US007818388B2

(12) **United States Patent**
Arimilli et al.

(10) **Patent No.: US 7,818,388 B2**
(45) **Date of Patent: Oct. 19, 2010**

(54) **DATA PROCESSING SYSTEM, METHOD AND INTERCONNECT FABRIC SUPPORTING MULTIPLE PLANES OF PROCESSING NODES**

(75) Inventors: **Ravi K. Arimilli**, Austin, TX (US); **Benjiman L. Goodman**, Cedar Park, TX (US); **Guy L. Guthrie**, Austin, TX (US); **Praveen S. Reddy**, Austin, TX (US); **William J. Starke**, Round Rock, TX (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 1412 days.

(21) Appl. No.: **11/245,887**

(22) Filed: **Oct. 7, 2005**

(65) **Prior Publication Data**

US 2007/0081516 A1 Apr. 12, 2007

(51) **Int. Cl.**
G06F 15/16 (2006.01)
(52) **U.S. Cl.** **709/208**
(58) **Field of Classification Search** 709/208
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,814,970	A	*	3/1989	Barbagelata et al.	712/28
4,855,903	A	*	8/1989	Carleton et al.	709/248
5,060,141	A		10/1991	Wakatani	
5,341,504	A		8/1994	Mori et al.	
5,392,446	A	*	2/1995	Tower et al.	712/11
5,590,345	A	*	12/1996	Barker et al.	712/11
5,659,785	A	*	8/1997	Pechanek et al.	712/11
5,842,031	A		11/1998	Barker et al.	
5,852,716	A		12/1998	Hagersten	
6,606,676	B1		8/2003	Deshpande et al.	
6,965,936	B1	*	11/2005	Wipfel et al.	709/224
7,096,323	B1		8/2006	Conway et al.	
7,162,590	B2		1/2007	Pruvost et al.	
2003/0009637	A1		1/2003	Arimilli et al.	
2003/0097529	A1		5/2003	Arimilli et al.	
2003/0154350	A1		8/2003	Edirisooriya et al.	
2006/0224833	A1		10/2006	Guthrie et al.	

* cited by examiner

OTHER PUBLICATIONS

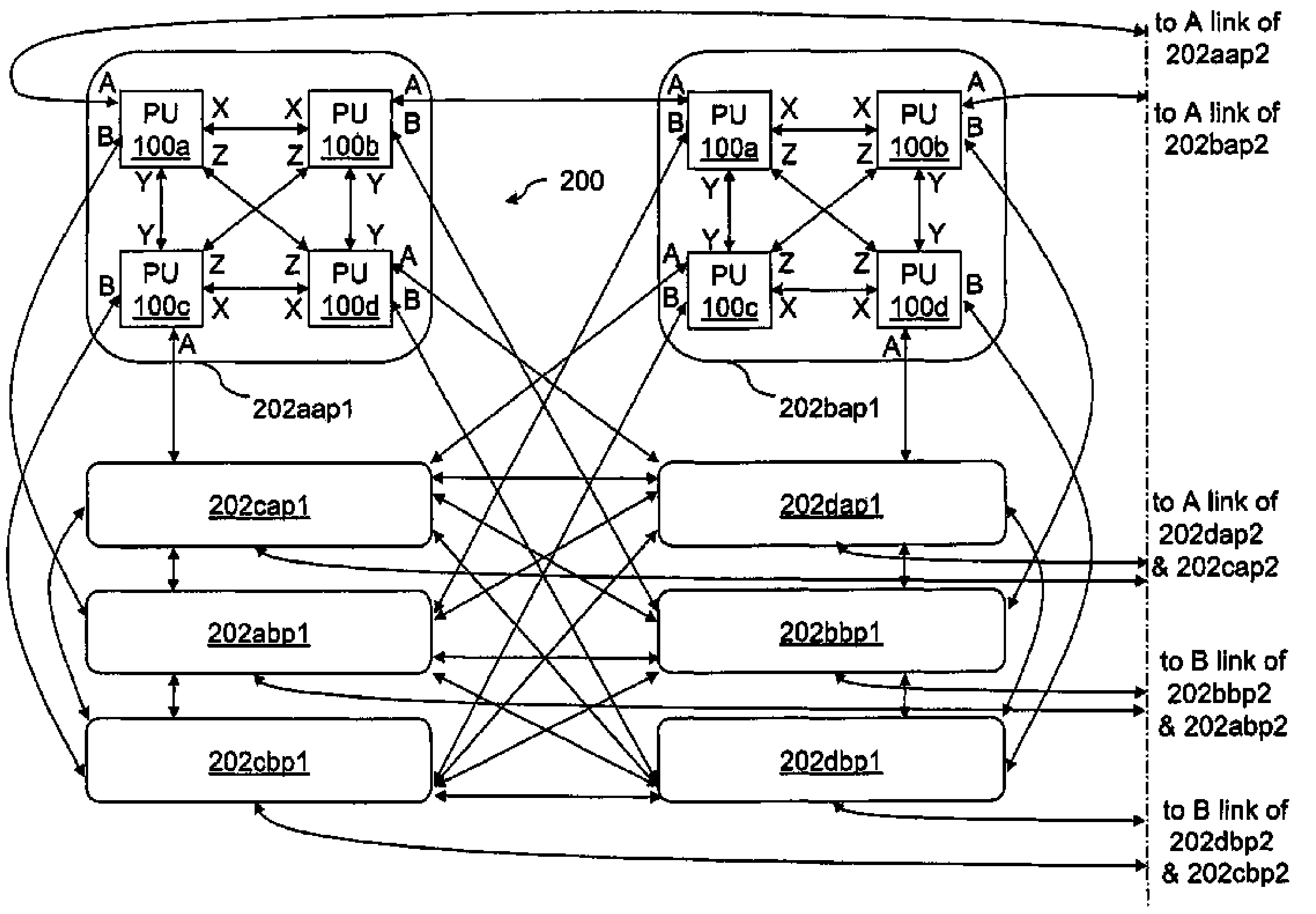
Patterson and Hennessy; "Computer Architecture-A Quantative Approach"; 1996; Morgan Kaufmann Publishers; 2^{nd} Edition; p. 75.

Primary Examiner—Shawki S Ismail
(74) *Attorney, Agent, or Firm*—Dillon & Yudell LLP

(57) **ABSTRACT**

A data processing system includes a first plane including a first plurality of processing nodes, each including multiple processing units, and a second plane including a second plurality of processing nodes, each including multiple processing units. The data processing system also includes a plurality of point-to-point first tier links. Each of the first plurality and second plurality of processing nodes includes one or more first tier links among the plurality of first tier links, where the first tier link(s) within each processing node connect a pair of processing units in the same processing node for communication. The data processing system further includes a plurality of point-to-point second tier links. At least a first of the plurality of second tier links connects processing units in different ones of the first plurality of processing nodes, at least a second of the plurality of second tier links connects processing units in different ones of the second plurality of processing nodes, and at least a third of the plurality of second tier links connects a processing unit in the first plane to a processing unit in the second plane.

17 Claims, 32 Drawing Sheets



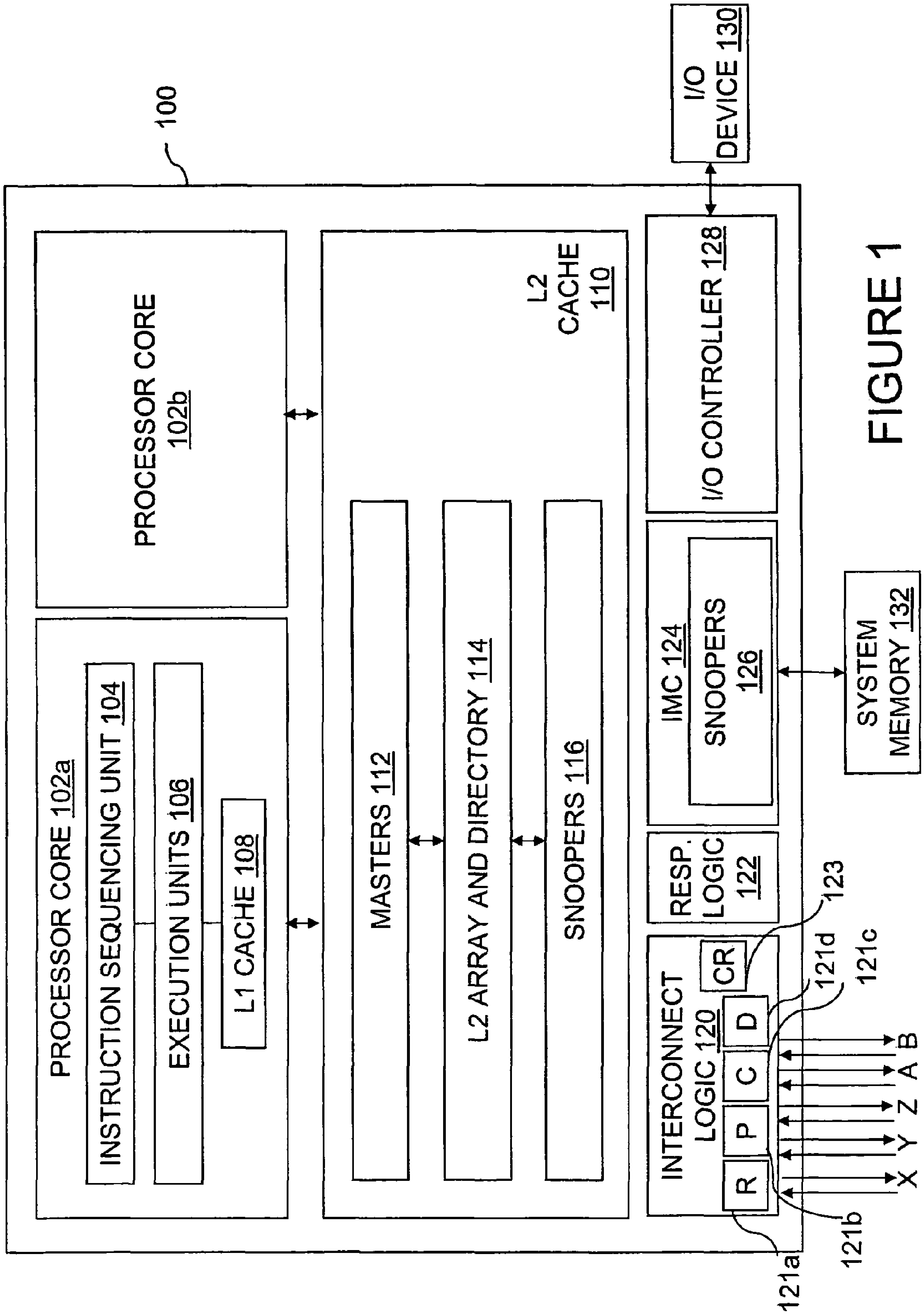

100
PROCESSOR CORE 102a
INSTRUCTION SEQUENCING UNIT 104
EXECUTION UNITS 106
L1 CACHE 108
PROCESSOR CORE 102b
L2 CACHE 110
MASTERS 112
L2 ARRAY AND DIRECTORY 114
SNOOPERS 116
INTERCONNECT LOGIC 120
R
P
C
D
CR
121a
121b
121c
121d
123
X
Y
Z
A
B
RESP. LOGIC 122
IMC 124
SNOOPERS 126
I/O CONTROLLER 128
I/O DEVICE 130
SYSTEM MEMORY 132

FIGURE 1

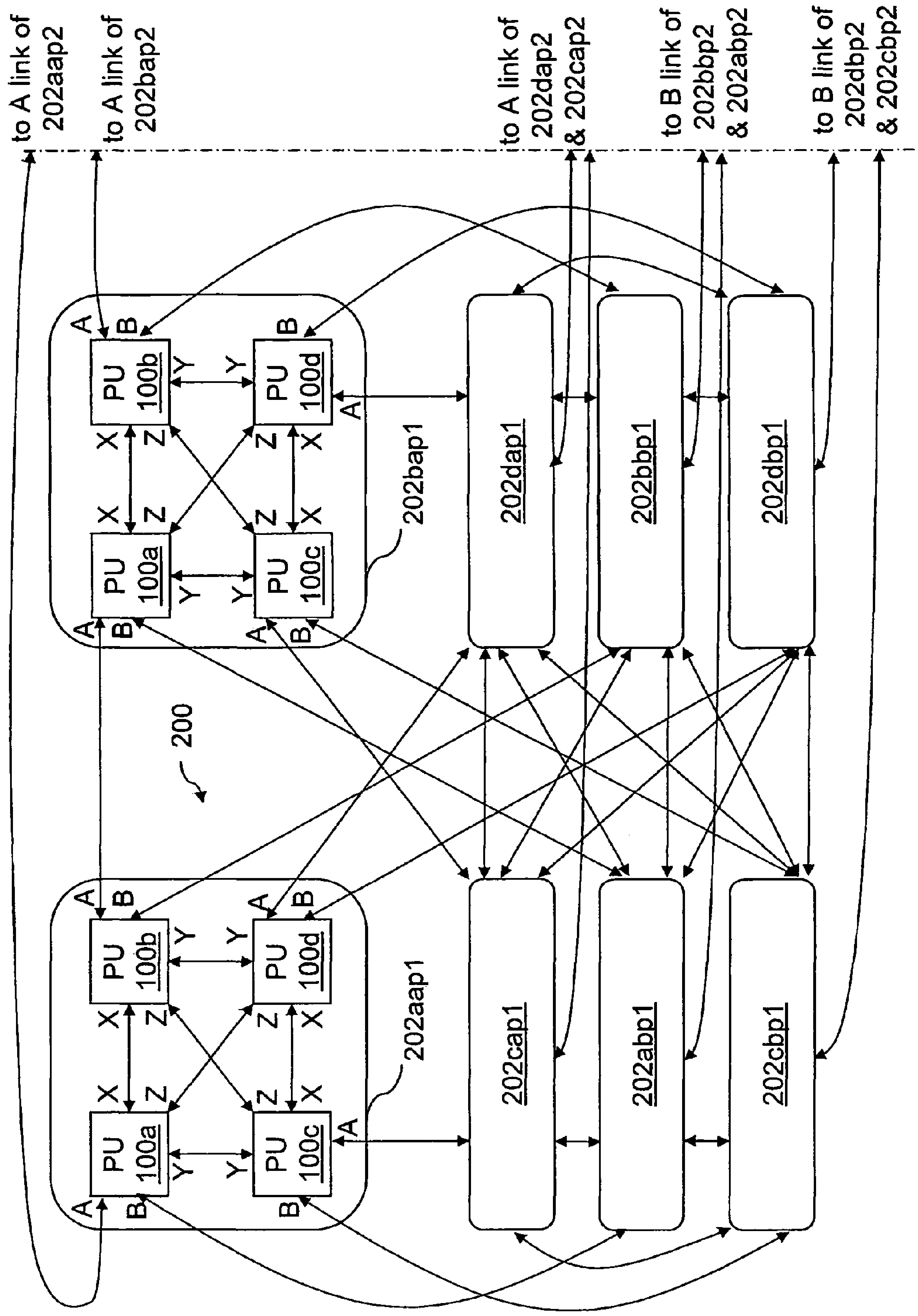

to A link of 202aap2
to A link of 202bap2
to A link of 202dap2 & 202cap2
to B link of 202bbp2 & 202abp2
to B link of 202dbp2 & 202cbp2
200
202bap1
PU 100a
PU 100b
PU 100c
PU 100d
A
B
X
Y
Z
202aap1
PU 100a
PU 100b
PU 100c
PU 100d
202dap1
202bbp1
202dbp1
202cap1
202abp1
202cbp1

FIGURE 2A

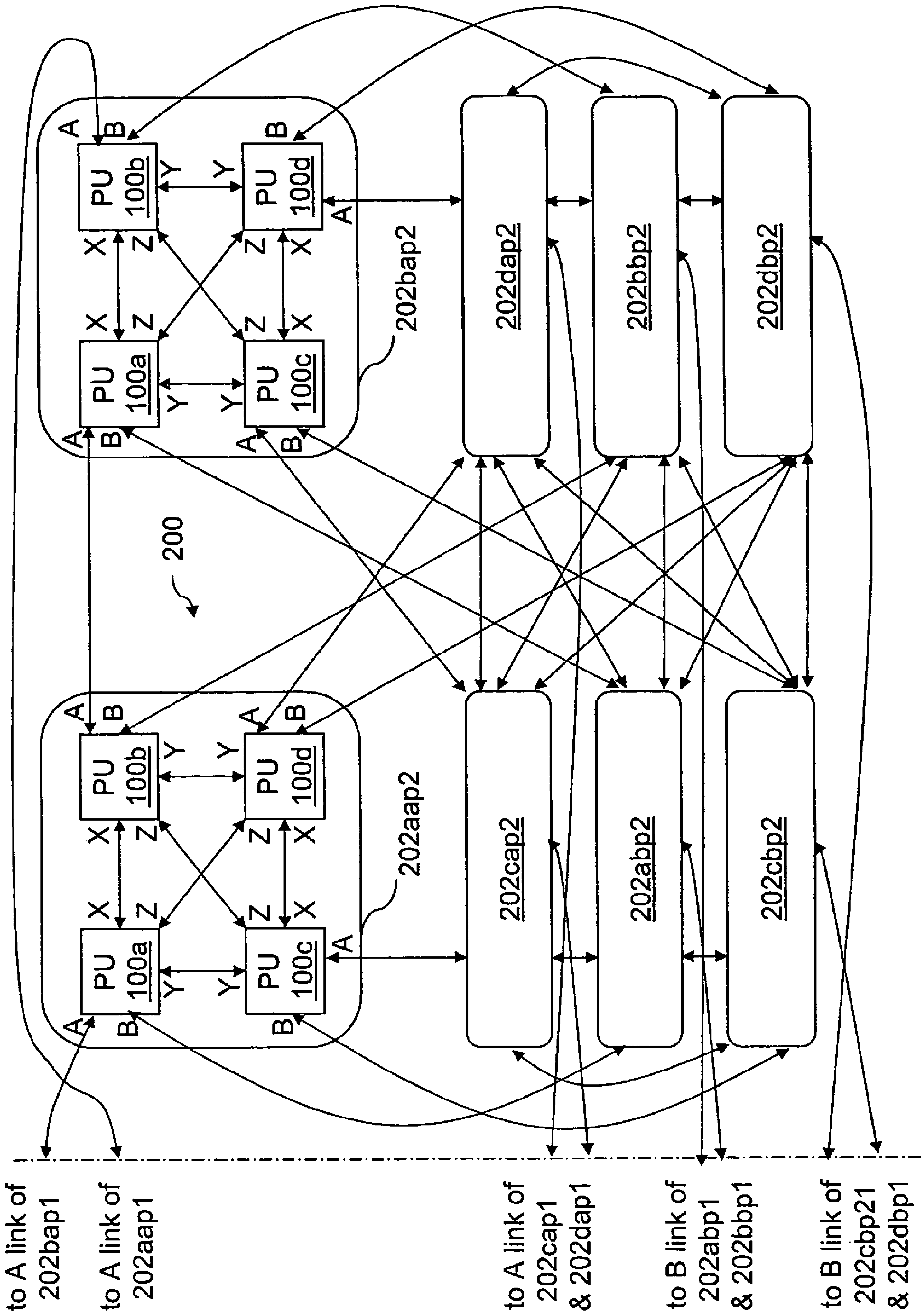

200
PU 100a
PU 100b
PU 100c
PU 100d
A
B
X
Y
Z
202aap2
202bap2
202cap2
202abp2
202cbp2
202dap2
202bbp2
202dbp2
to A link of 202bap1
to A link of 202aap1
to A link of 202cap1 & 202dap1
to B link of 202abp1 & 202bbp1
to B link of 202cbp21 & 202dbp1

FIGURE 2B

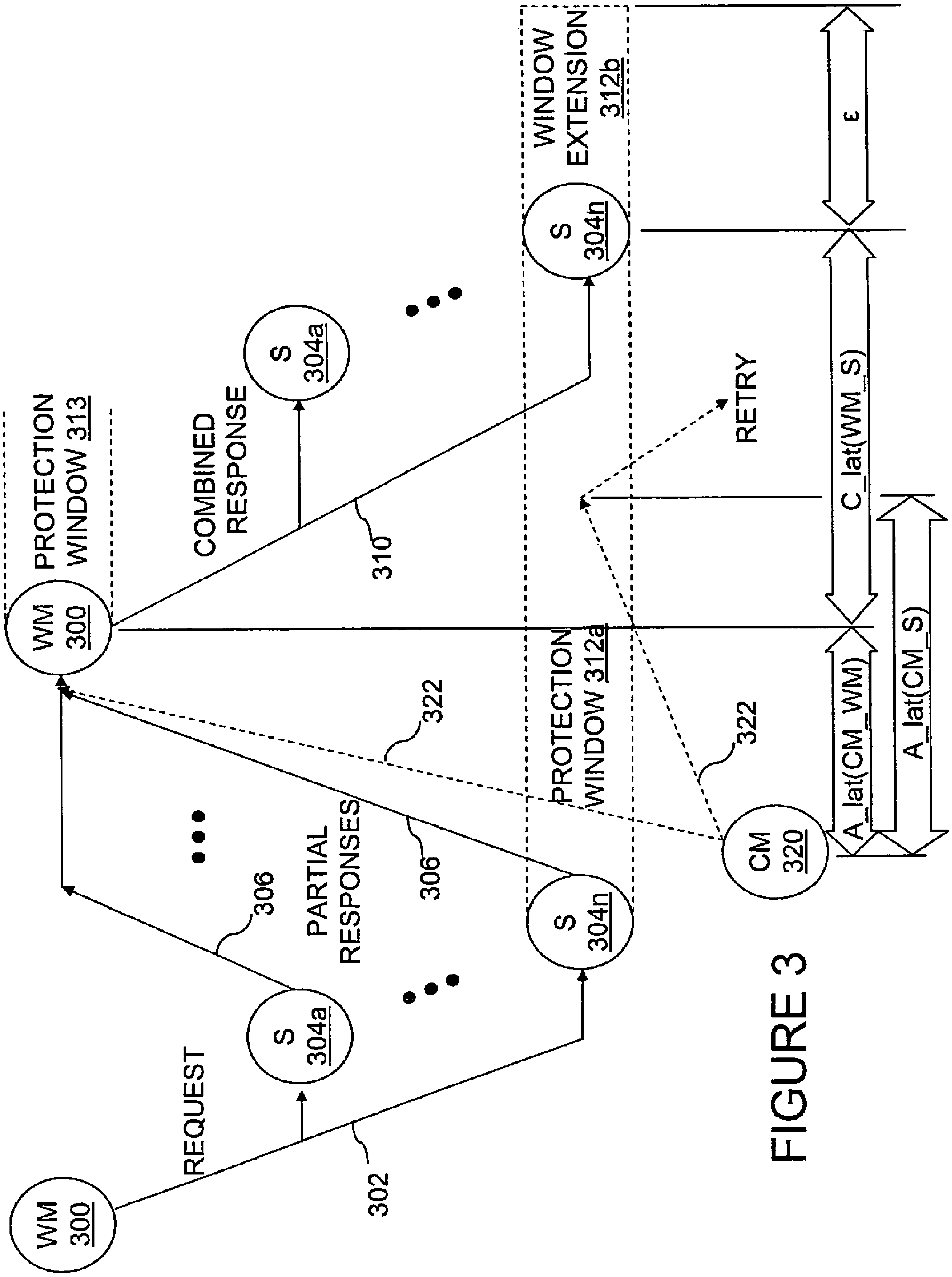

WM 300
REQUEST
302
S 304a
S 304n
306
PARTIAL RESPONSES
306
PROTECTION WINDOW 312a
322
CM 320
322
WM 300
PROTECTION WINDOW 313
COMBINED RESPONSE
310
S 304a
RETRY
S 304n
WINDOW EXTENSION 312b
A_lat(CM_WM)
A_lat(CM_S)
C_lat(WM_S)
ε

FIGURE 3

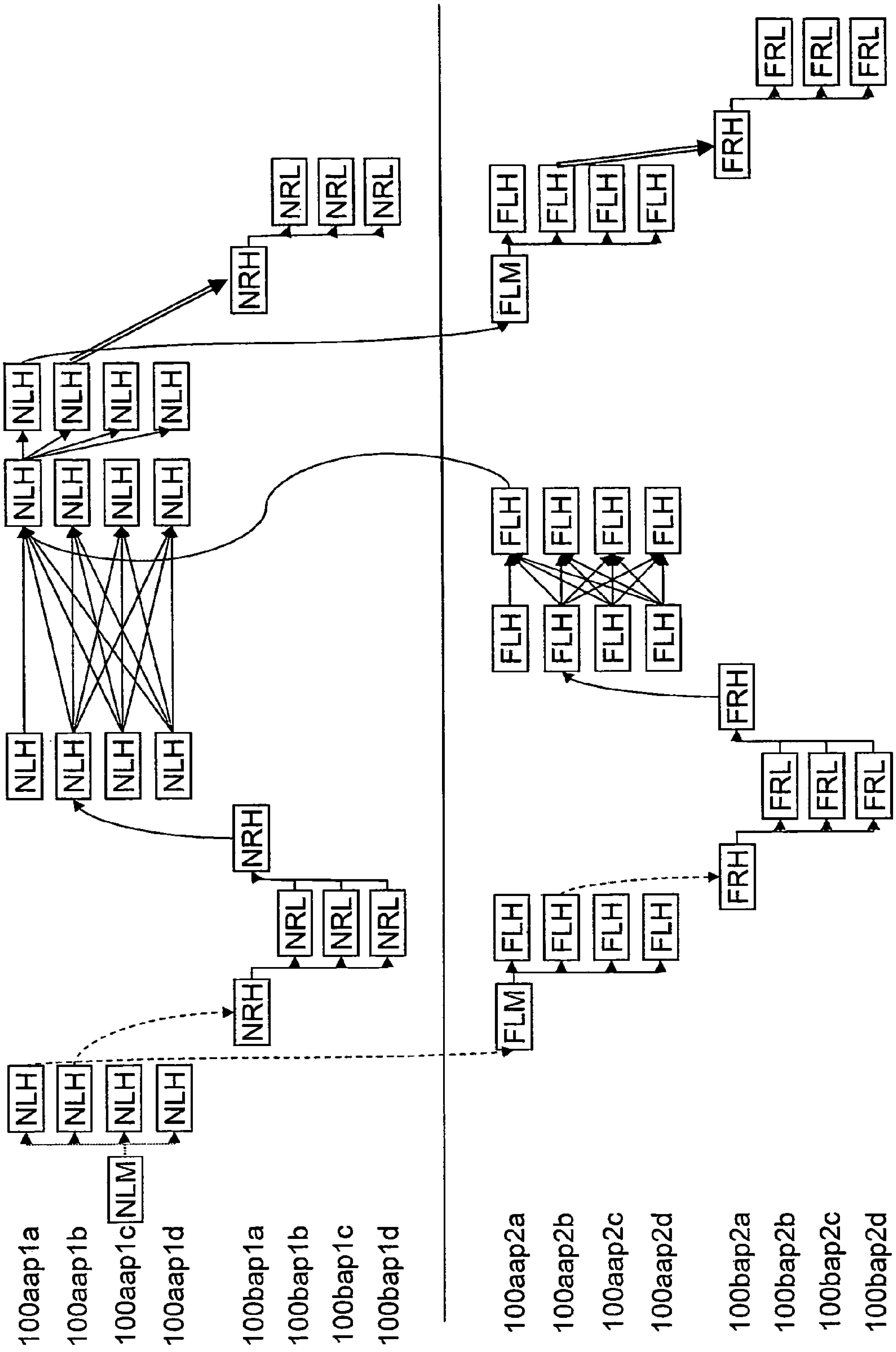

100aap1a
100aap1b
100aap1c
100aap1d
100bap1a
100bap1b
100bap1c
100bap1d
100aap2a
100aap2b
100aap2c
100aap2d
100bap2a
100bap2b
100bap2c
100bap2d
NLM
NLH
NRH
NRL
FLM
FLH
FRH
FRL

FIGURE 4A

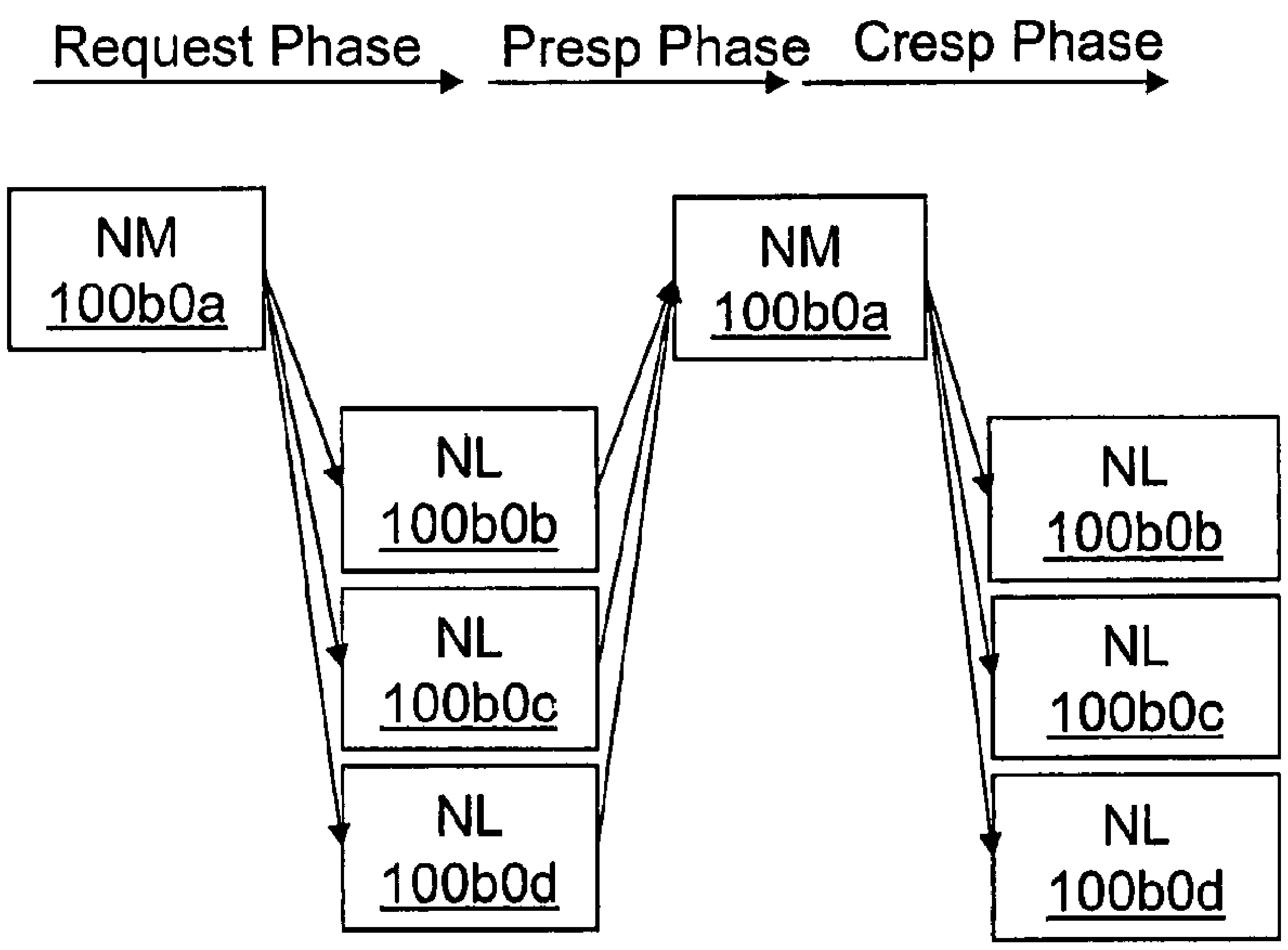


FIGURE 4B

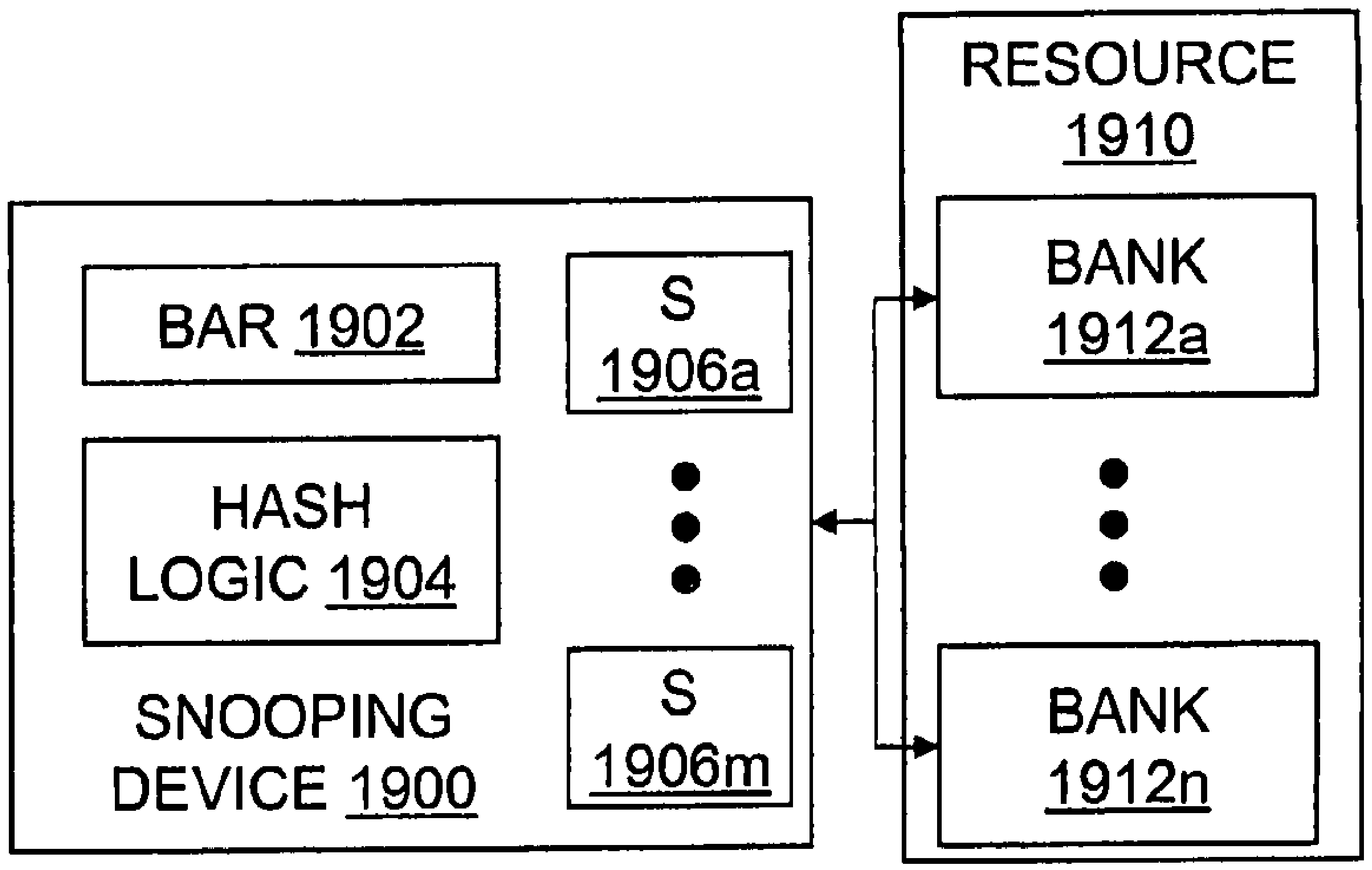


FIGURE 19

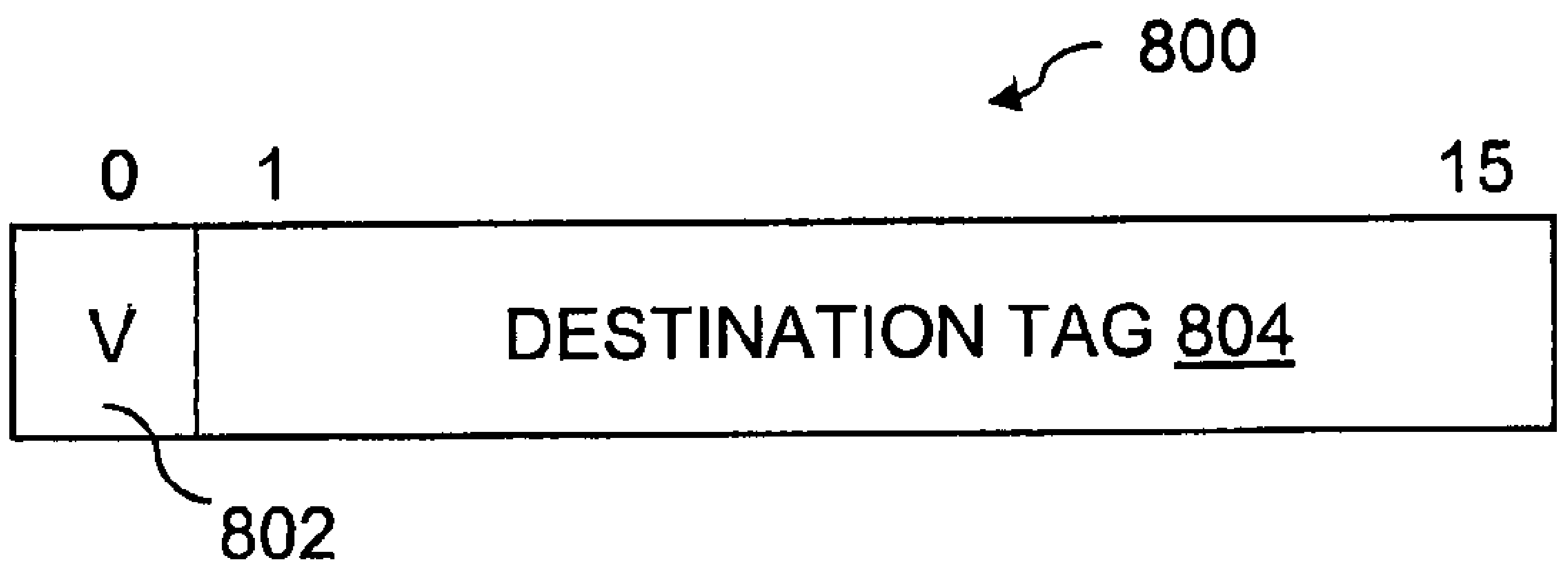


FIGURE 8

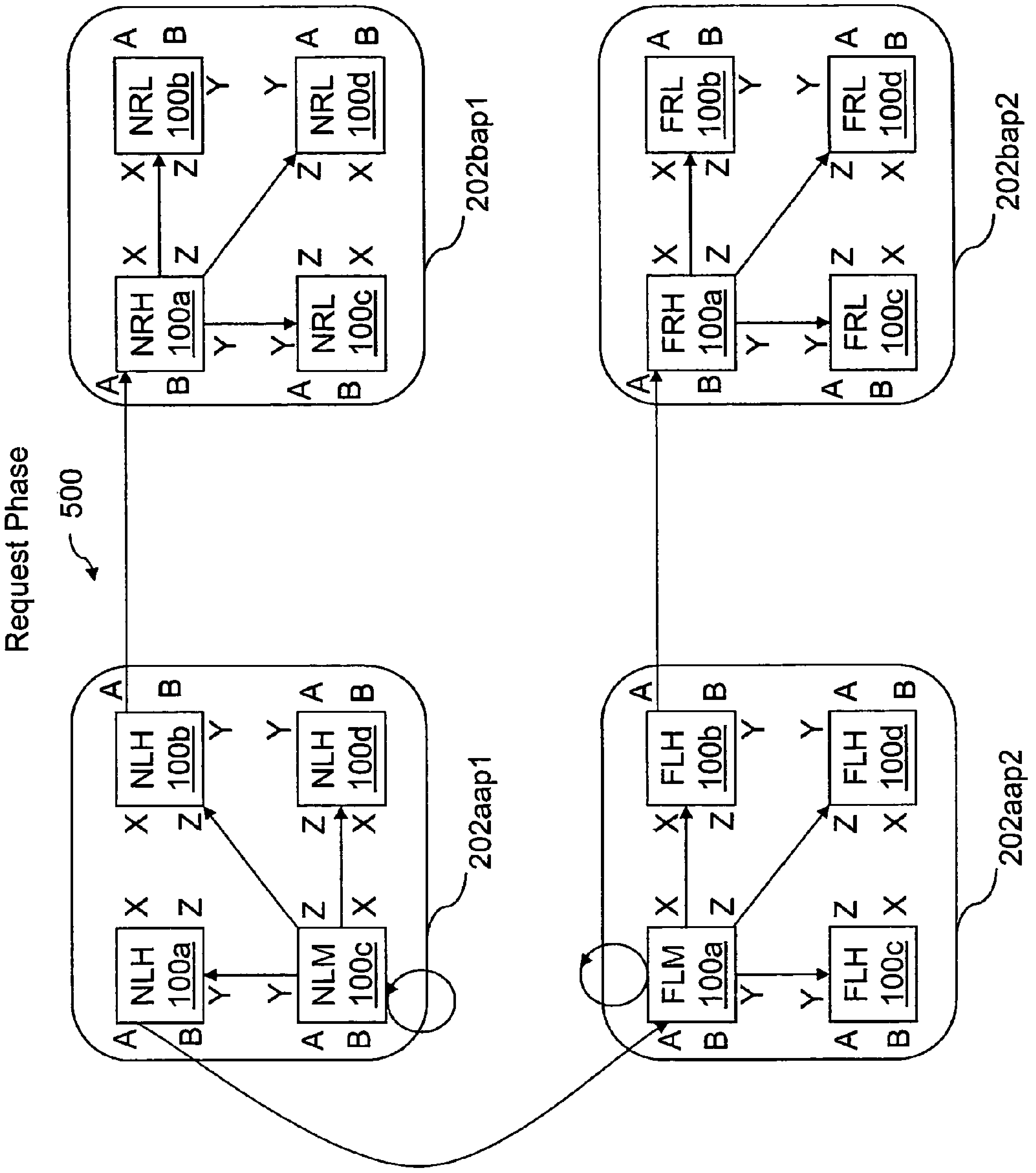

Request Phase
500
NLH 100a
NLM 100c
NLH 100b
NLH 100d
202aap1
FLM 100a
FLH 100c
FLH 100b
FLH 100d
202aap2
NRH 100a
NRL 100c
NRL 100b
NRL 100d
202bap1
FRH 100a
FRL 100c
FRL 100b
FRL 100d
202bap2
A
B
X
Y
Z

FIGURE 5A

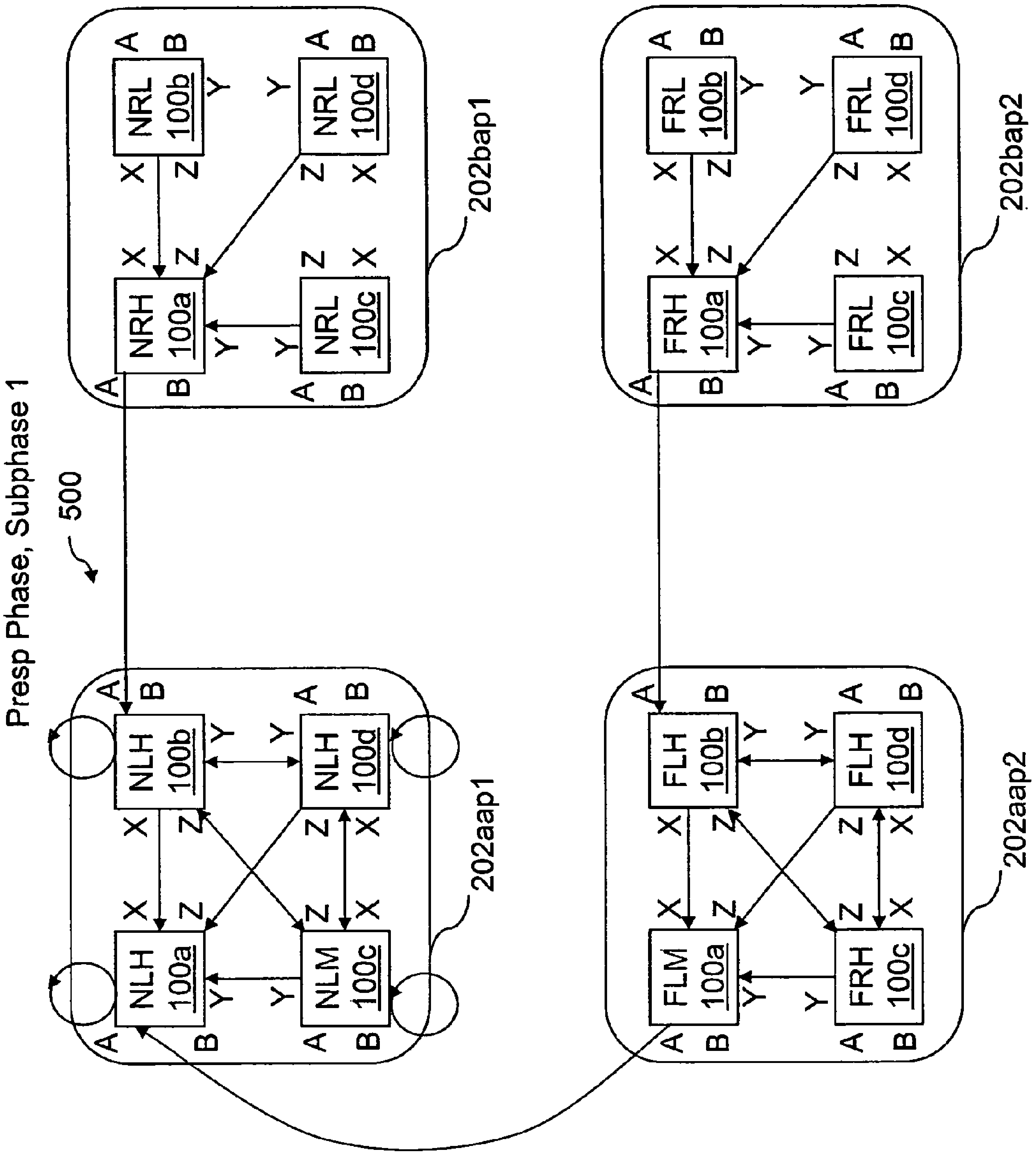

Presp Phase, Subphase 1
500
NLH 100a
NLH 100b
NLM 100c
NLH 100d
202aap1
NRH 100a
NRL 100b
NRL 100c
NRL 100d
202bap1
FLM 100a
FLH 100b
FRH 100c
FLH 100d
202aap2
FRH 100a
FRL 100b
FRL 100c
FRL 100d
202bap2
A
B
X
Y
Z

FIGURE 5B

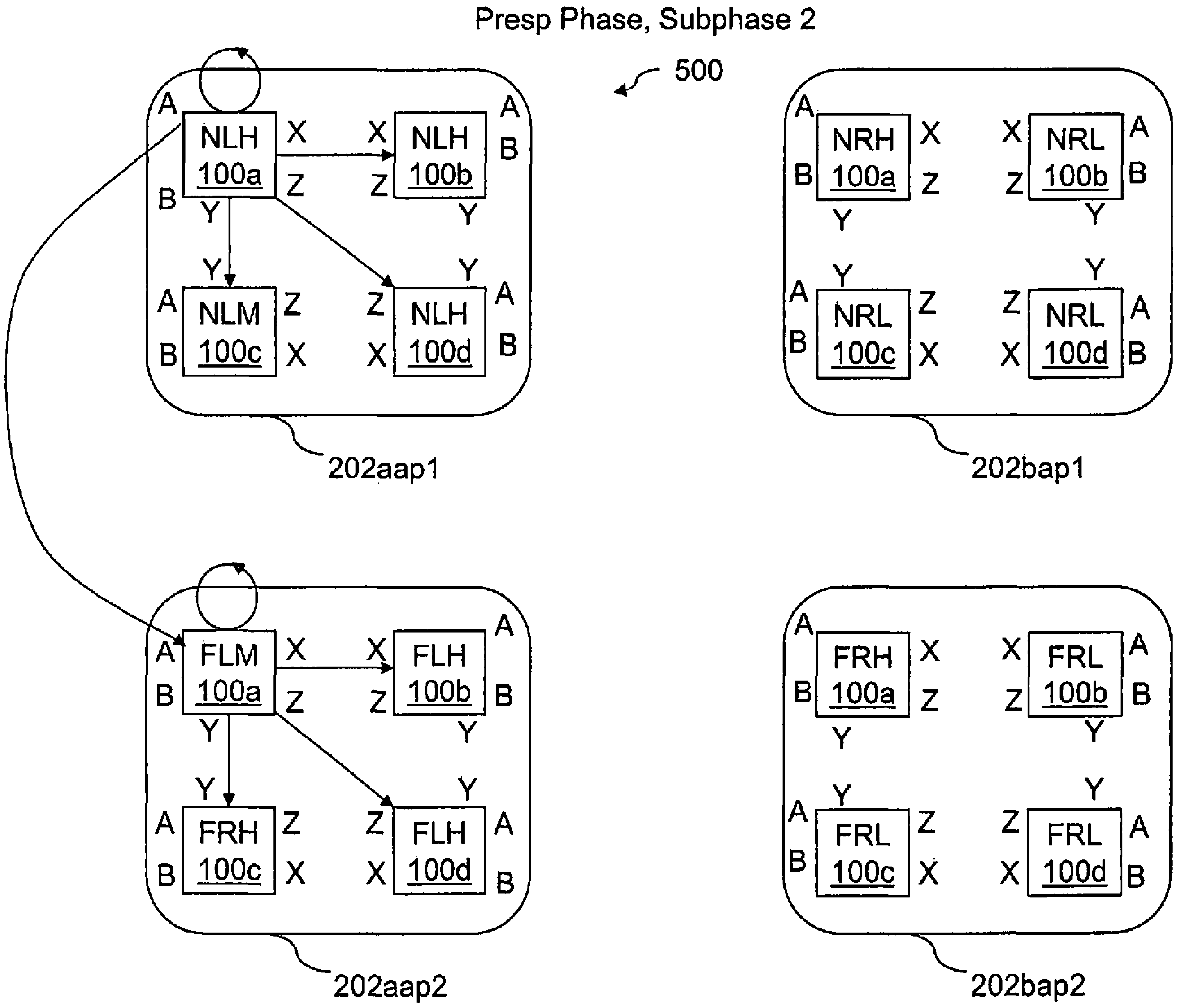

Presp Phase, Subphase 2
500
A
B
NLH
100a
X
Z
Y
X
Z
NLH
100b
Y
A
B
Y
A
B
NLM
100c
Z
X
Y
Z
X
NLH
100d
A
B
202aap1
A
B
NRH
100a
X
Z
Y
X
Z
NRL
100b
Y
A
B
Y
A
B
NRL
100c
Z
X
Y
Z
X
NRL
100d
A
B
202bap1
A
B
FLM
100a
X
Z
Y
X
Z
FLH
100b
Y
A
B
Y
A
B
FRH
100c
Z
X
Y
Z
X
FLH
100d
A
B
202aap2
A
B
FRH
100a
X
Z
Y
X
Z
FRL
100b
Y
A
B
Y
A
B
FRL
100c
Z
X
Y
Z
X
FRL
100d
A
B
202bap2

FIGURE 5C

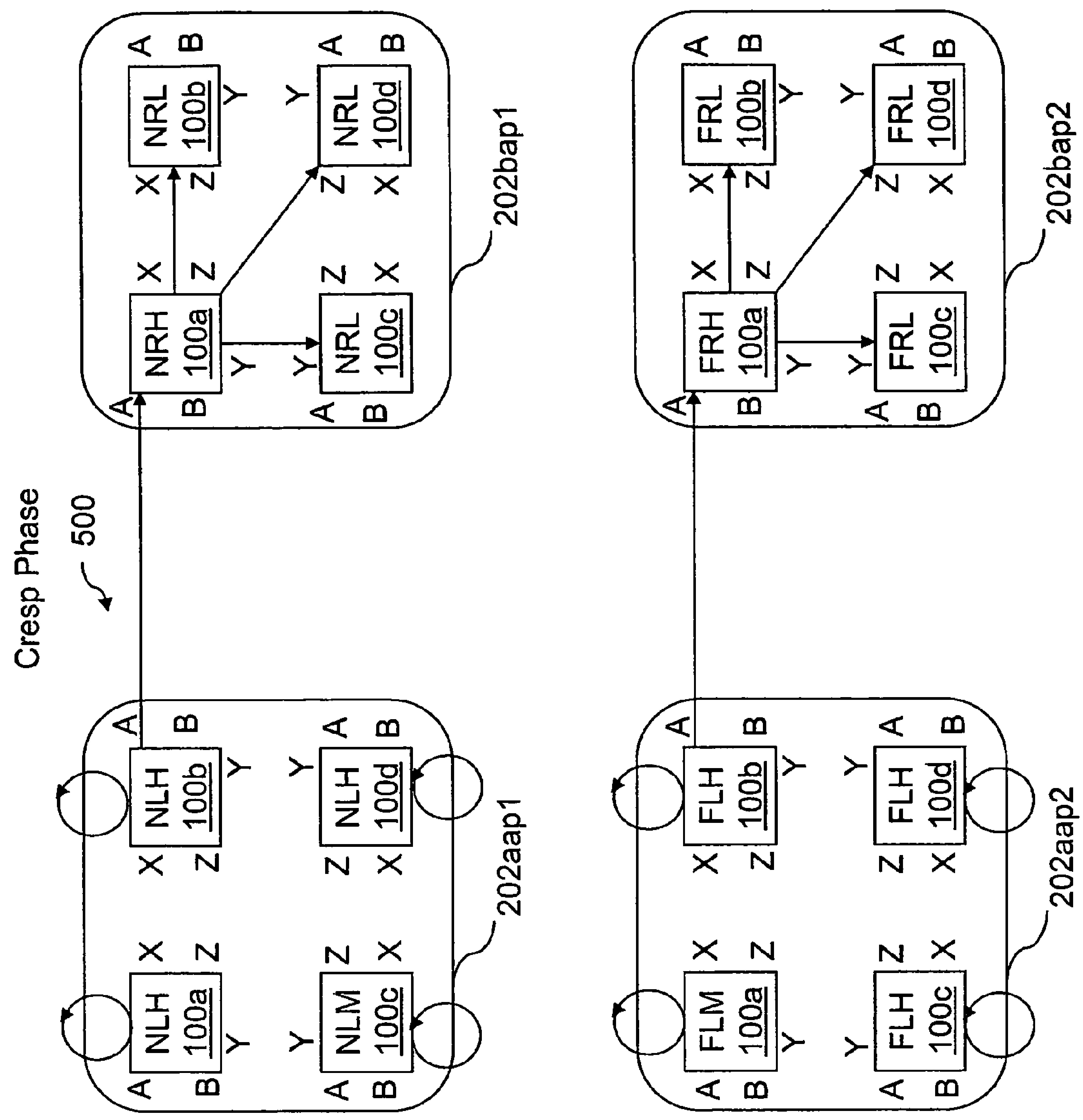

Cresp Phase
500
NLH 100a
NLH 100b
NLM 100c
NLH 100d
202aap1
NRH 100a
NRL 100b
NRL 100c
NRL 100d
202bap1
FLM 100a
FLH 100b
FLH 100c
FLH 100d
202aap2
FRH 100a
FRL 100b
FRL 100c
FRL 100d
202bap2
A
B
X
Y
Z

FIGURE 5D

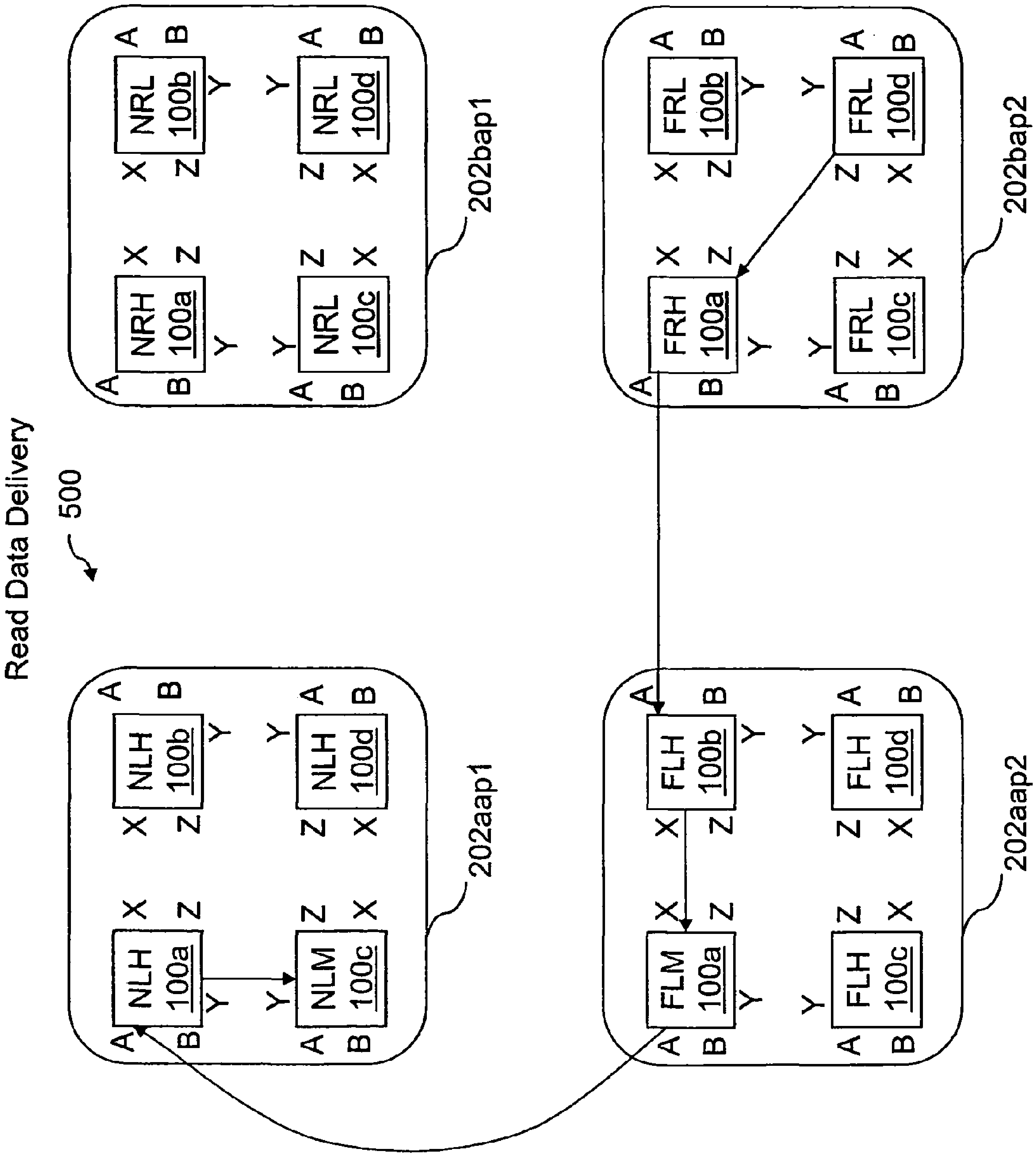

Read Data Delivery
500
NLH 100a
NLH 100b
NLM 100c
NLH 100d
202aap1
NRH 100a
NRL 100b
NRL 100c
NRL 100d
202bap1
FLM 100a
FLH 100b
FLH 100c
FLH 100d
202aap2
FRH 100a
FRL 100b
FRL 100c
FRL 100d
202bap2
A
B
X
Y
Z

FIGURE 6A

Write Data Delivery

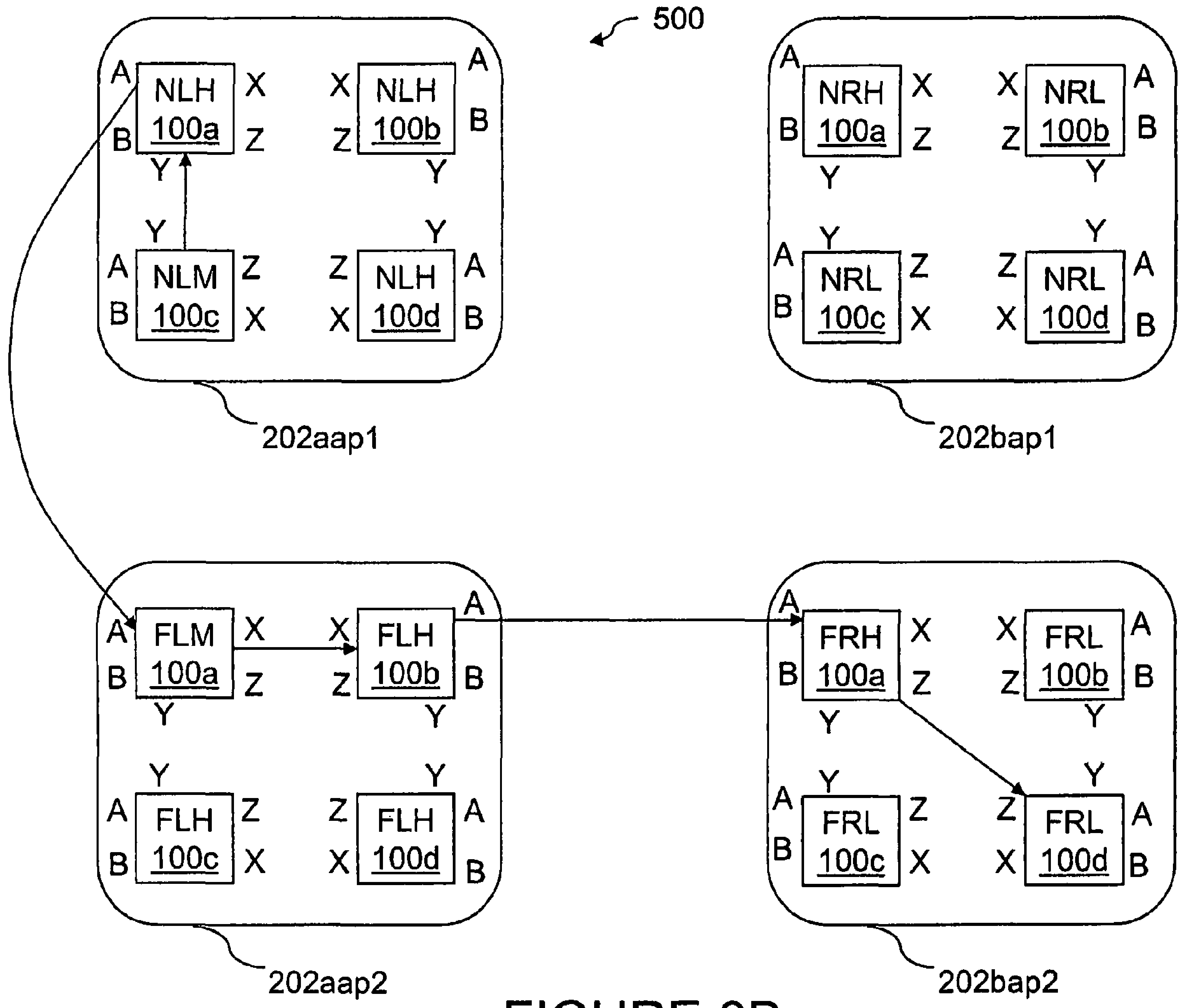
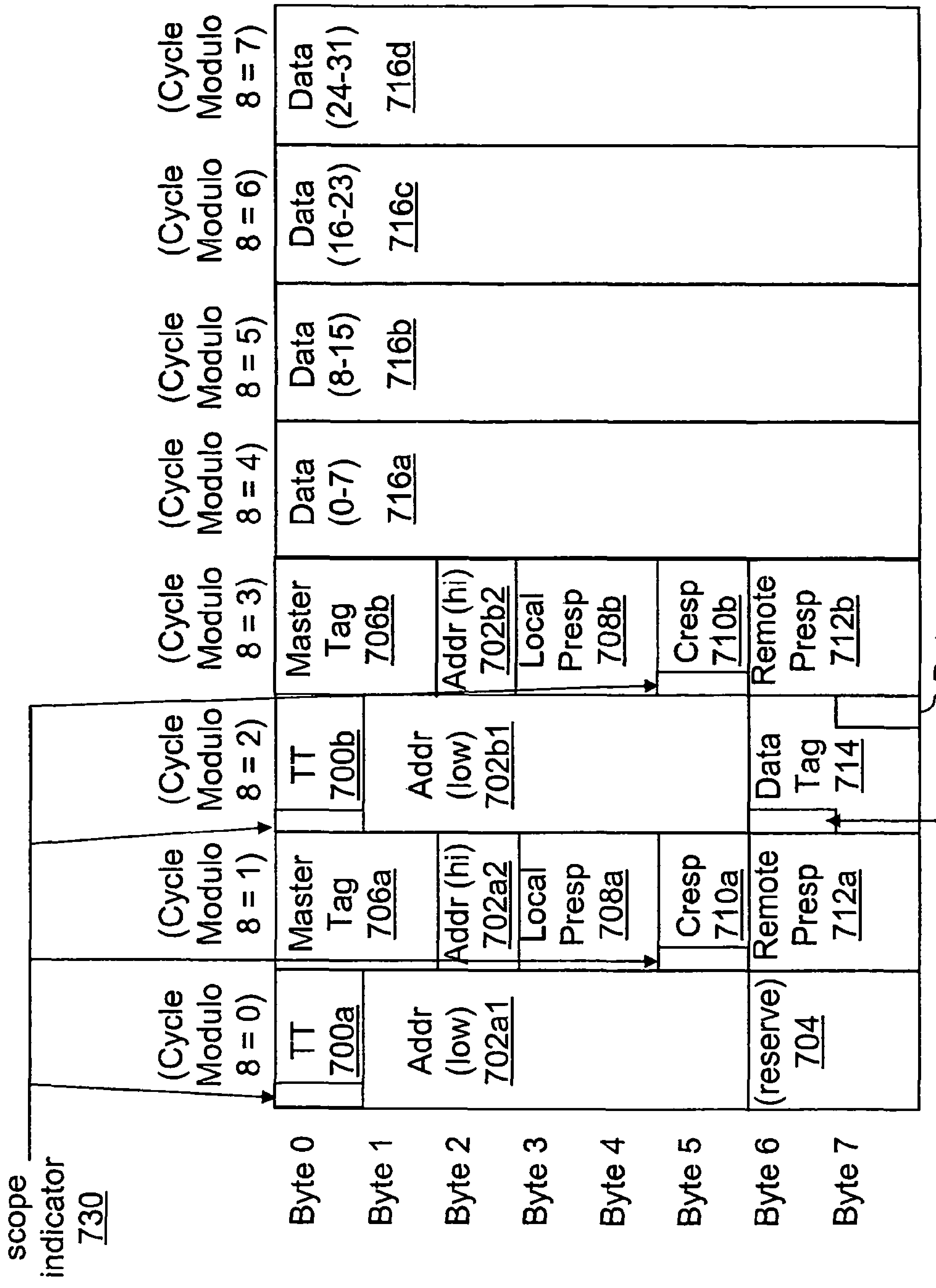
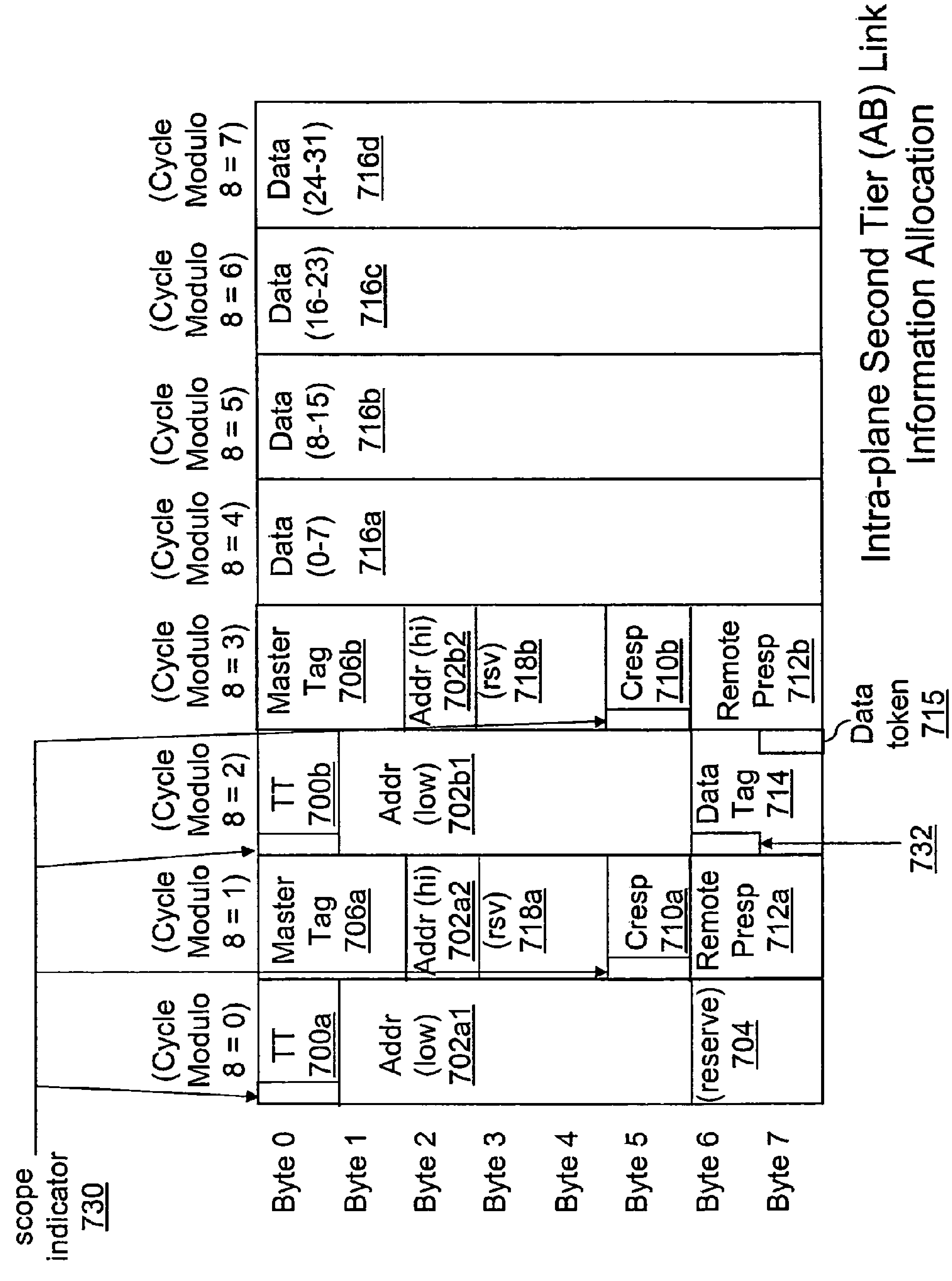


FIGURE 6B



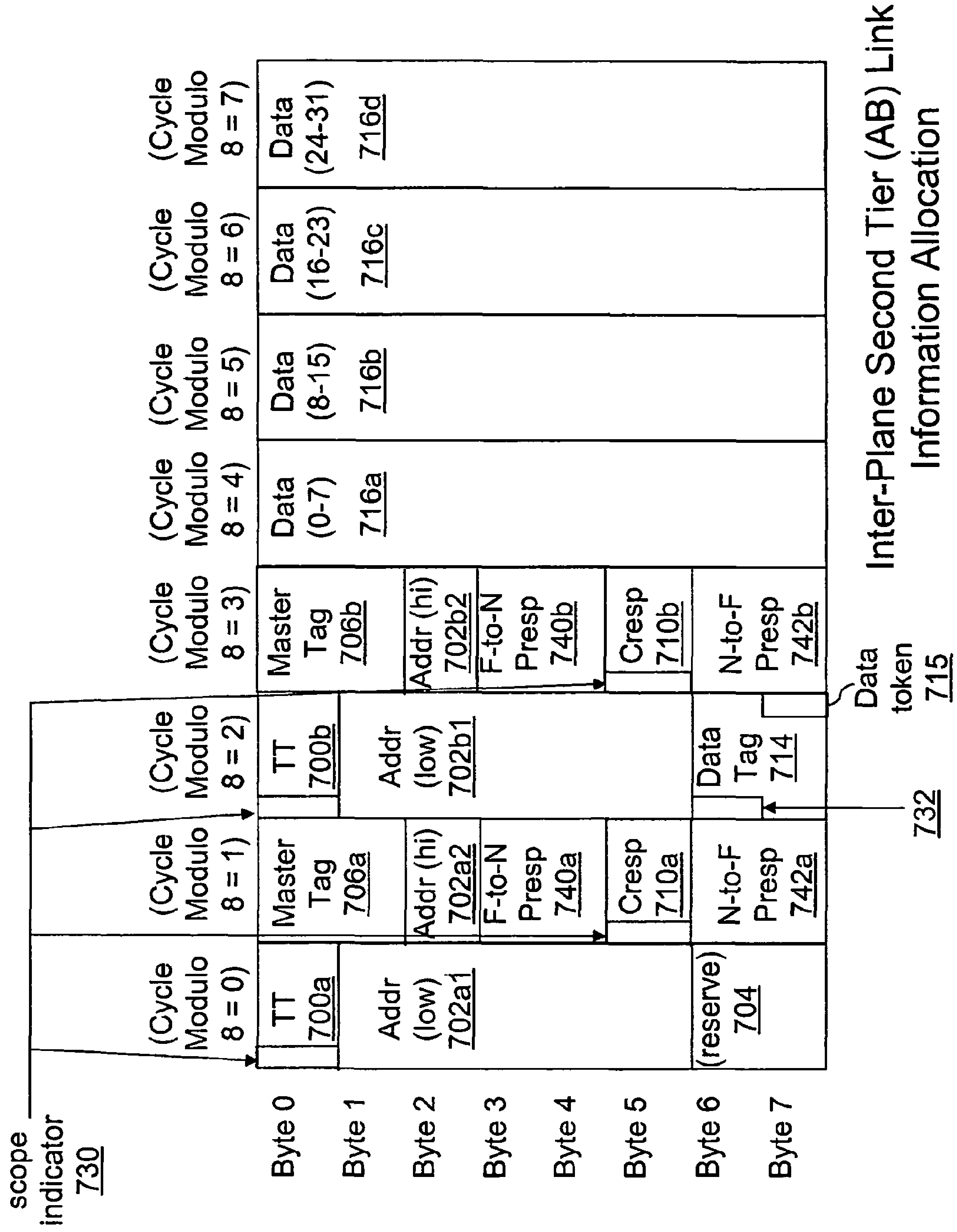
First Tier (XYZ) Link Information Allocation

FIGURE 7A



Intra-plane Second Tier (AB) Link Information Allocation

FIGURE 7B



Inter-Plane Second Tier (AB) Link Information Allocation

FIGURE 7C

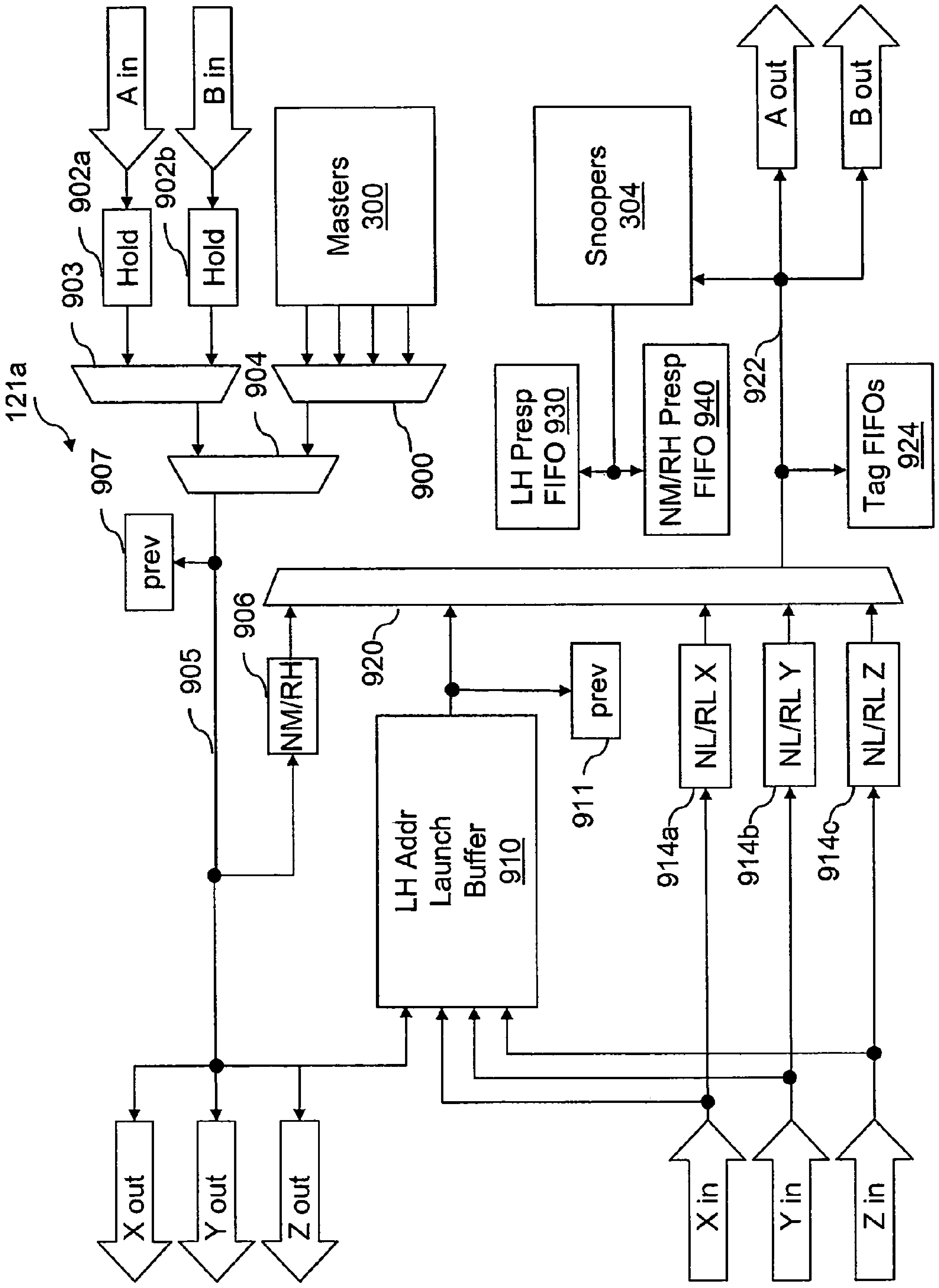

121a
A in
B in
902a
902b
903
Hold
Hold
904
Masters
300
900
907
prev
906
NM/RH
905
920
LH Addr
Launch
Buffer
910
prev
911
NL/RL X
NL/RL Y
NL/RL Z
914a
914b
914c
X out
Y out
Z out
X in
Y in
Z in
Snoopers
304
A out
B out
LH Presp
FIFO 930
NM/RH Presp
FIFO 940
922
Tag FIFOs
924

FIGURE 9

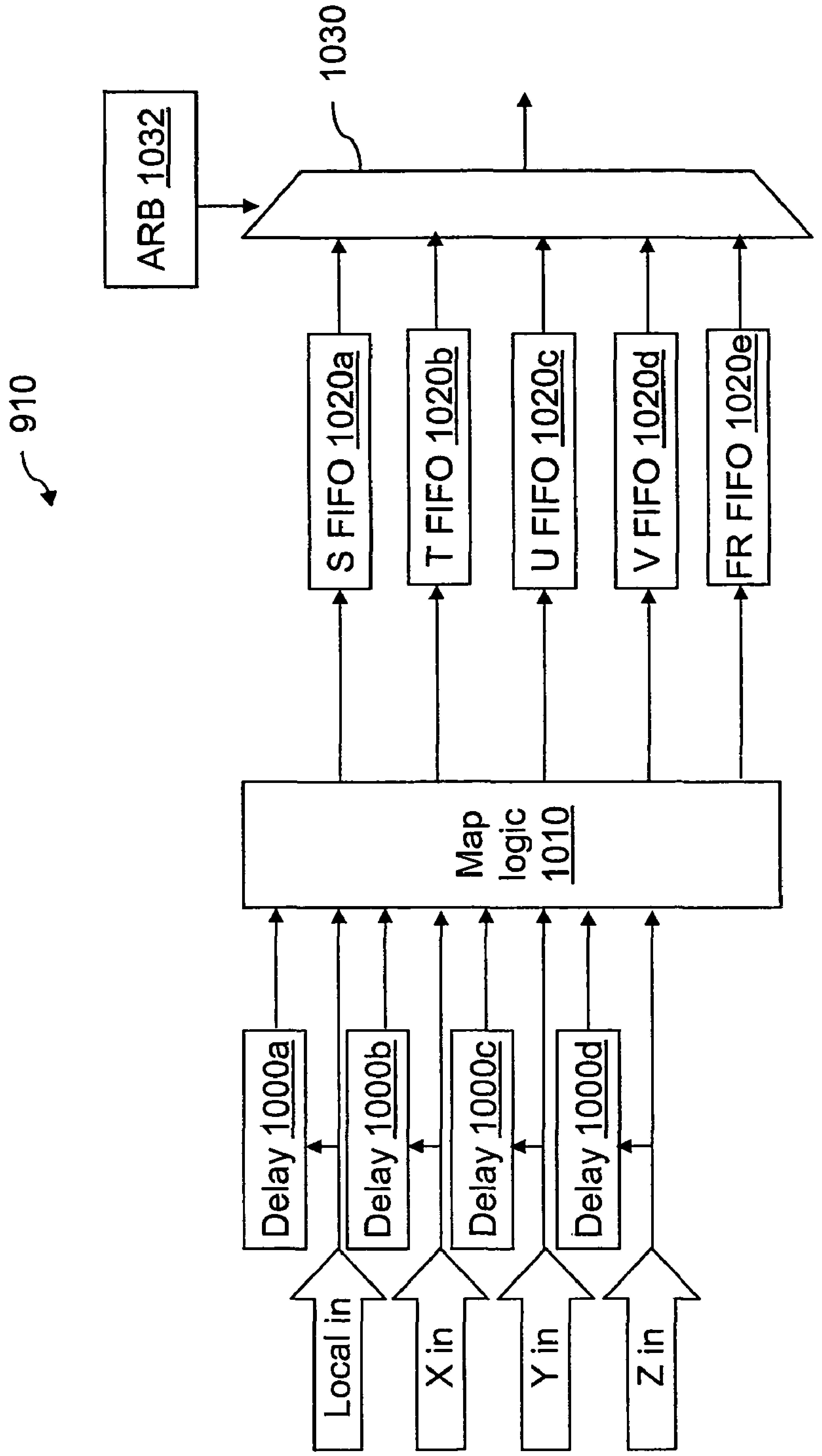

910
ARB 1032
1030
S FIFO 1020a
T FIFO 1020b
U FIFO 1020c
V FIFO 1020d
FR FIFO 1020e
Map logic 1010
Delay 1000a
Delay 1000b
Delay 1000c
Delay 1000d
Local in
X in
Y in
Z in

FIGURE 10

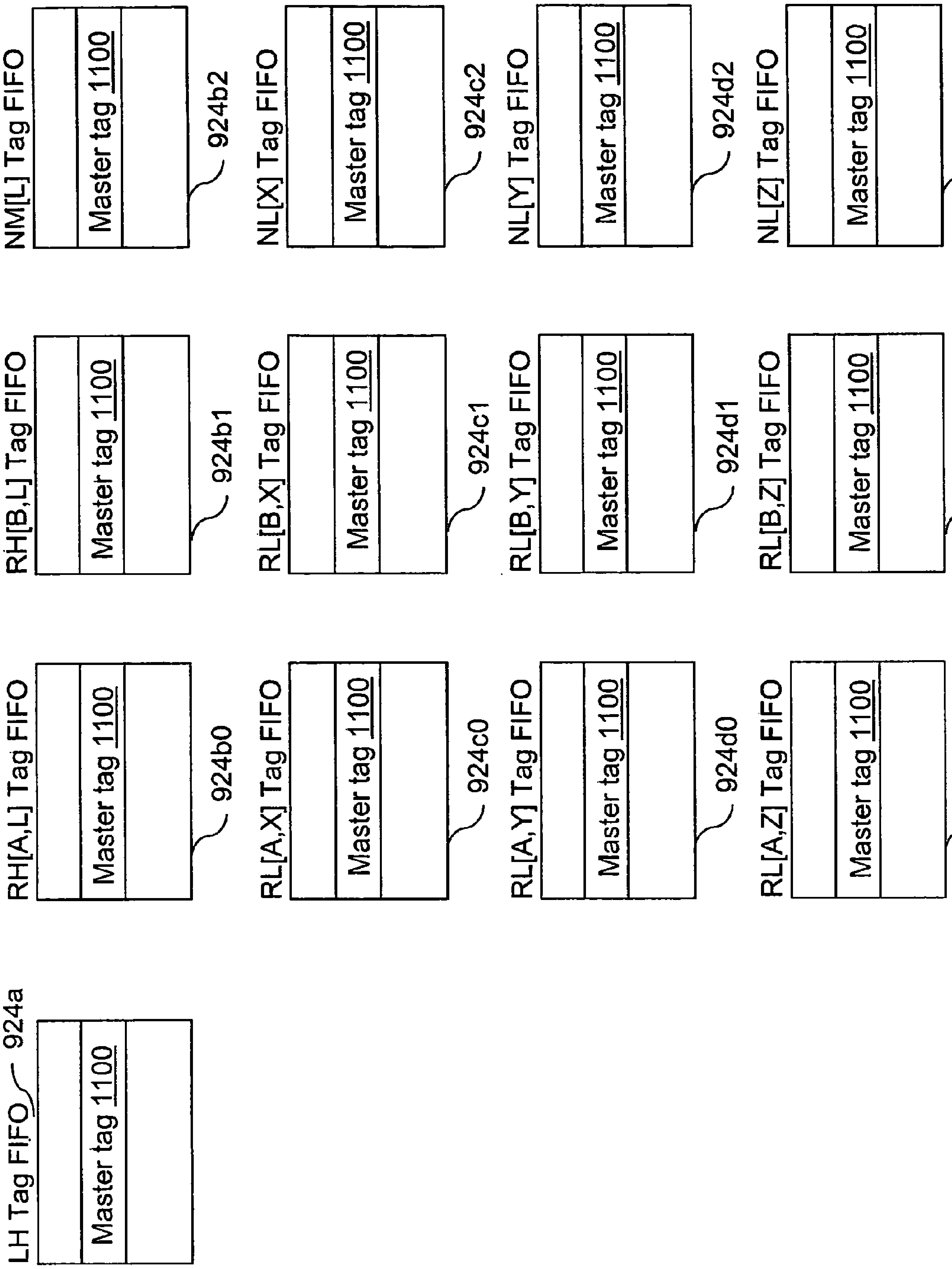

LH Tag FIFO
924a
Master tag 1100
RH[A,L] Tag FIFO
Master tag 1100
924b0
RH[B,L] Tag FIFO
Master tag 1100
924b1
NM[L] Tag FIFO
Master tag 1100
924b2
RL[A,X] Tag FIFO
Master tag 1100
924c0
RL[B,X] Tag FIFO
Master tag 1100
924c1
NL[X] Tag FIFO
Master tag 1100
924c2
RL[A,Y] Tag FIFO
Master tag 1100
924d0
RL[B,Y] Tag FIFO
Master tag 1100
924d1
NL[Y] Tag FIFO
Master tag 1100
924d2
RL[A,Z] Tag FIFO
Master tag 1100
924e0
RL[B,Z] Tag FIFO
Master tag 1100
924e1
NL[Z] Tag FIFO
Master tag 1100
924e2

FIGURE 11

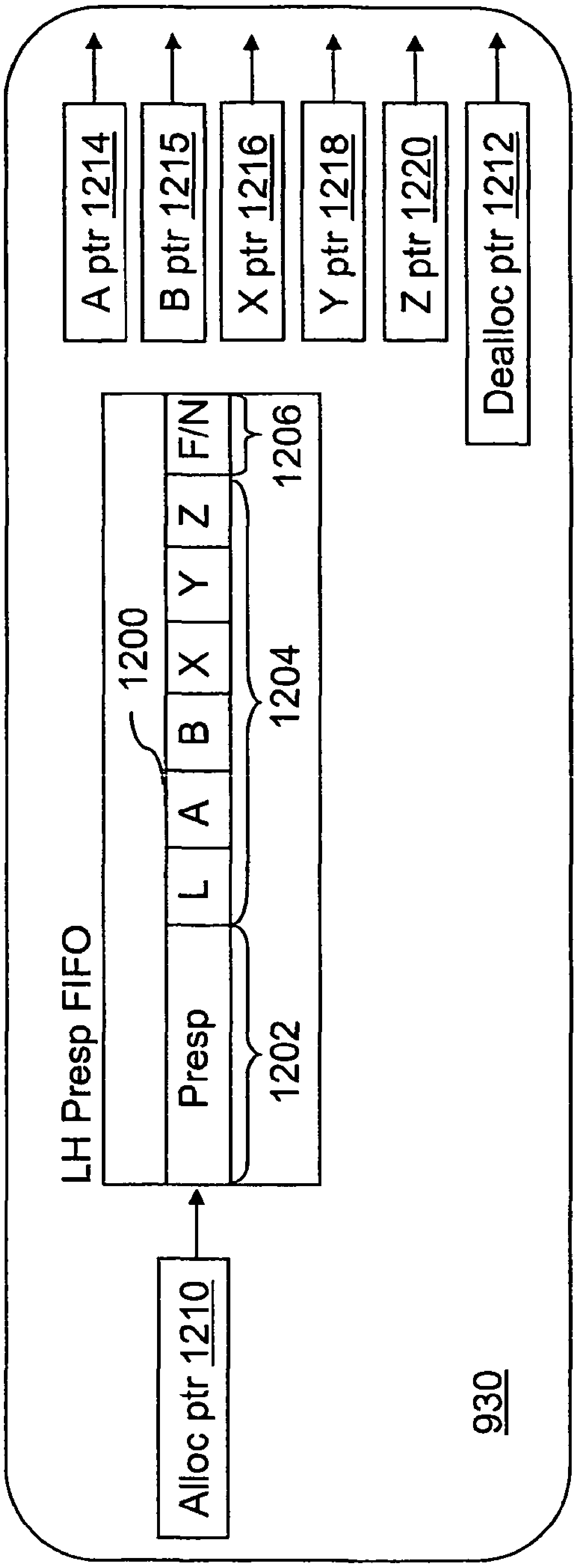


FIGURE 12A

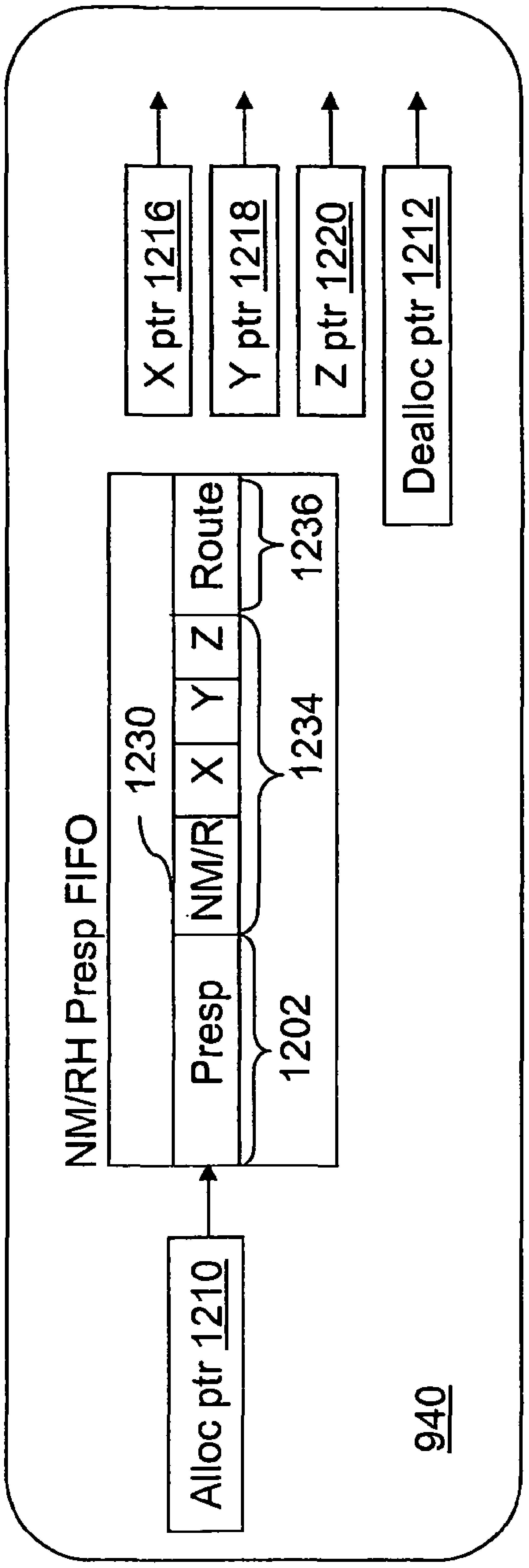


FIGURE 12B

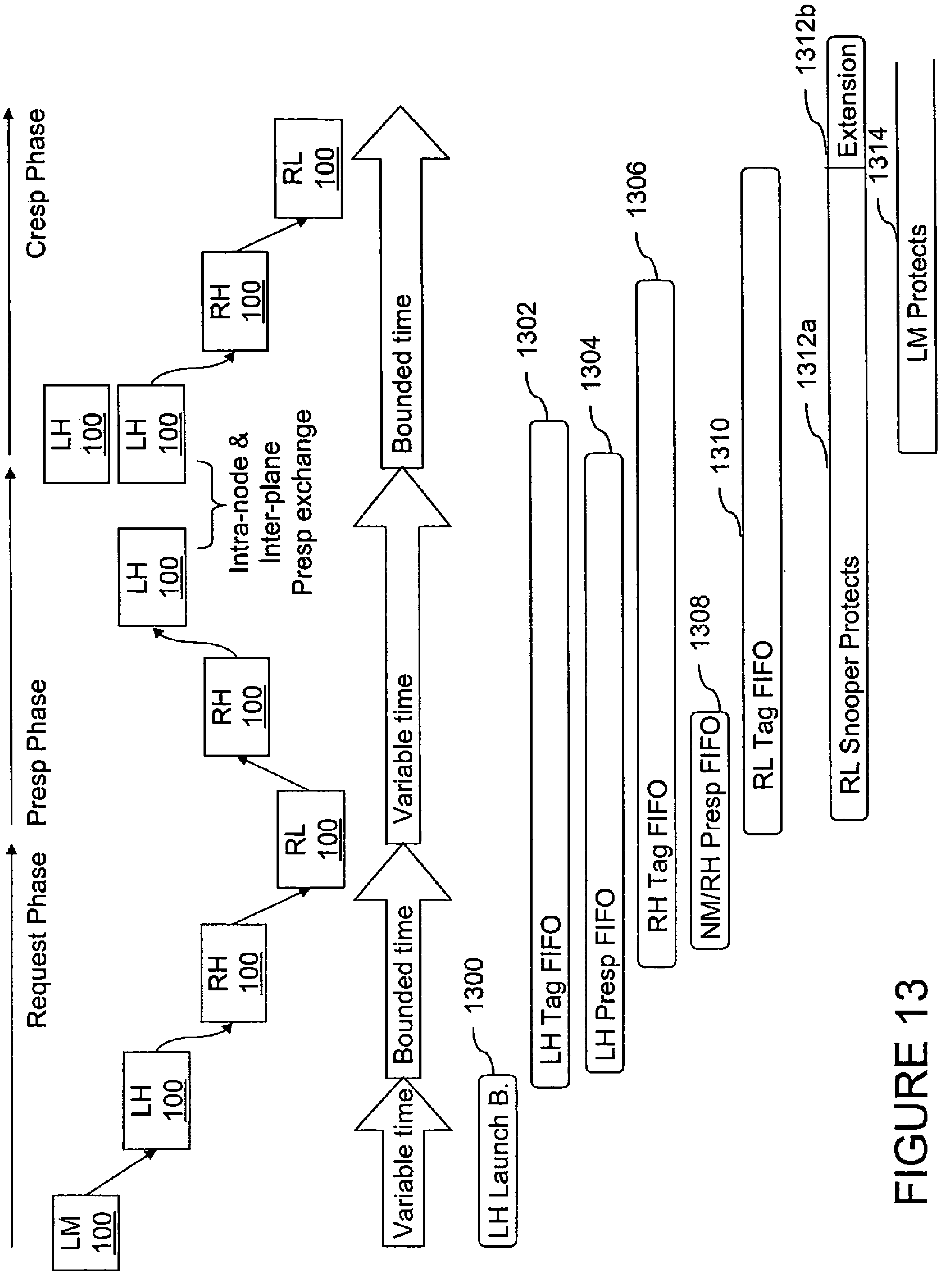

Request Phase
Presp Phase
Cresp Phase
LM 100
LH 100
RH 100
RL 100
RH 100
LH 100
Intra-node & Inter-plane Presp exchange
LH 100
LH 100
RH 100
RL 100
Variable time
Bounded time
Variable time
Bounded time
LH Launch B.
1300
LH Tag FIFO
1302
LH Presp FIFO
1304
RH Tag FIFO
1306
NM/RH Presp FIFO
1308
RL Tag FIFO
1310
RL Snooper Protects
1312a
Extension
1312b
LM Protects
1314

FIGURE 13

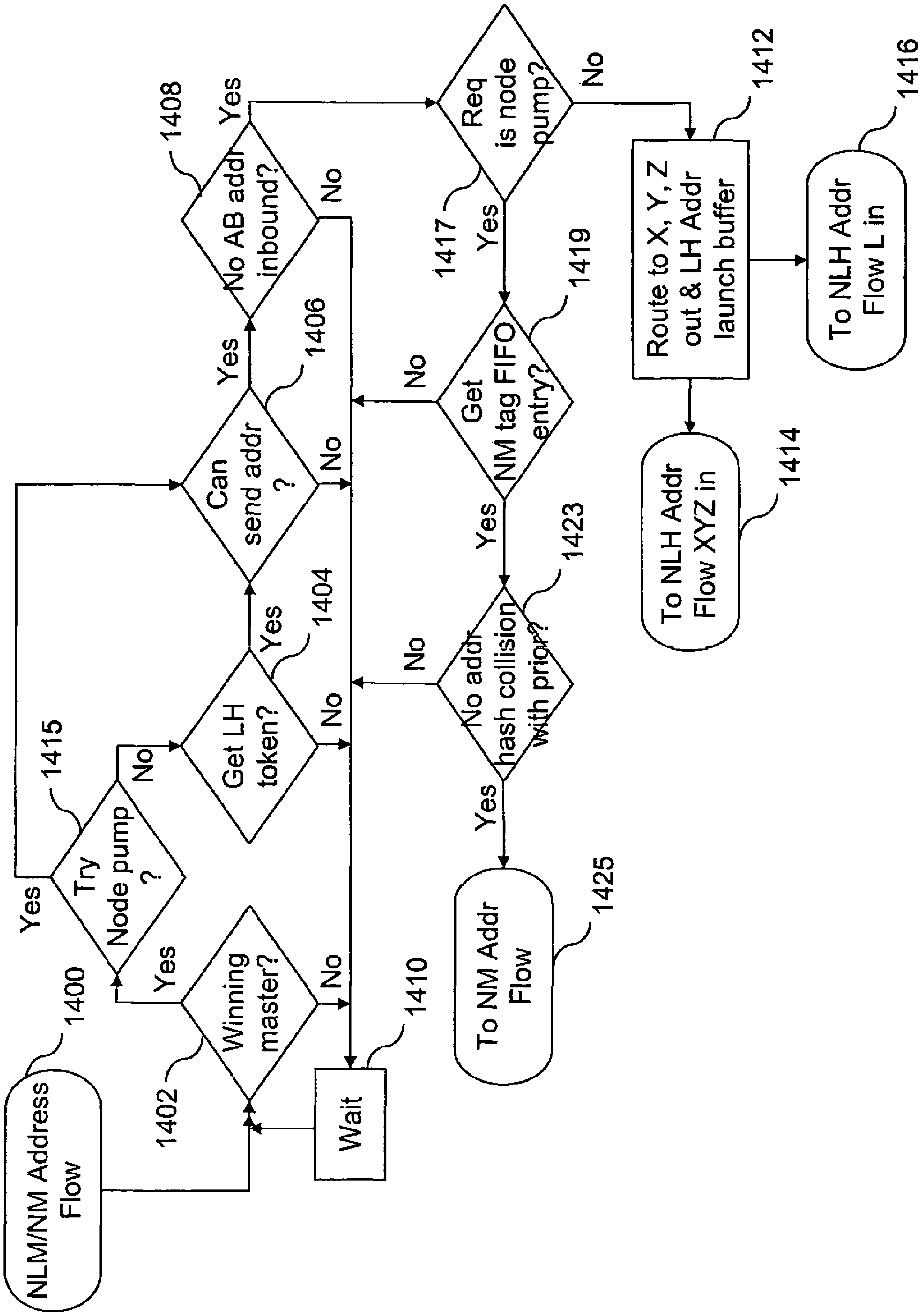

NLM/NM Address Flow
1400
Winning master?
1402
Yes
No
Try Node pump ?
1415
Yes
No
Get LH token?
1404
Yes
No
Can send addr ?
1406
Yes
No
No AB addr inbound?
1408
Yes
No
Wait
1410
Req is node pump?
1417
Yes
No
Get NM tag FIFO entry?
1419
Yes
No
No addr hash collision with prior?
1423
Yes
No
To NM Addr Flow
1425
Route to X, Y, Z out & LH Addr launch buffer
1412
To NLH Addr Flow XYZ in
1414
To NLH Addr Flow L in
1416

FIGURE 14A

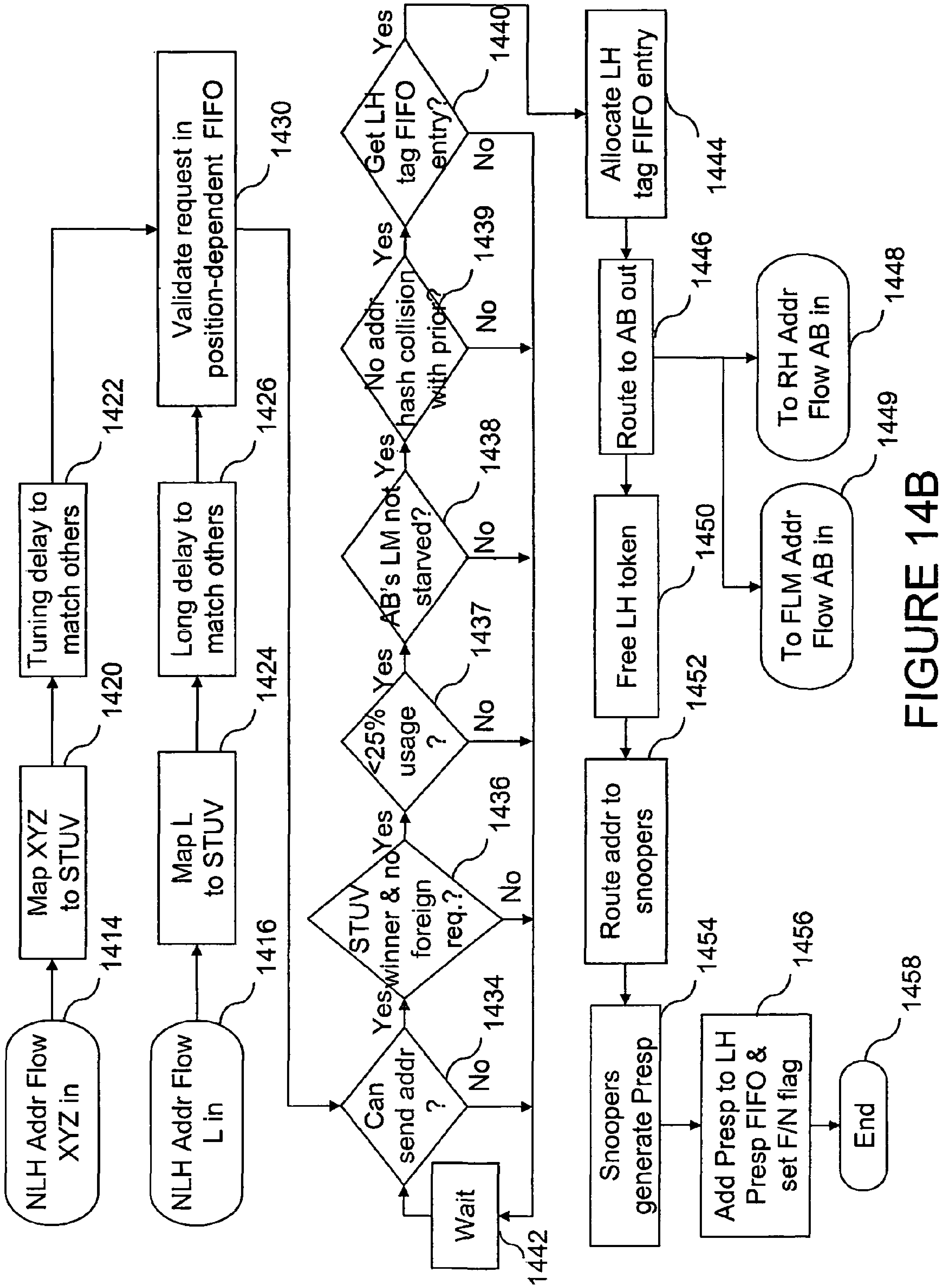

NLH Addr Flow XYZ in
1414
Map XYZ to STUV
1420
Tuning delay to match others
1422
NLH Addr Flow L in
1416
Map L to STUV
1424
Long delay to match others
1426
Validate request in position-dependent FIFO
1430
Can send addr ?
1434
Yes
No
STUV winner & no foreign req.?
1436
Yes
No
<25% usage ?
1437
Yes
No
AB's LM not starved?
1438
Yes
No
No addr hash collision with prior?
1439
Yes
No
Get LH tag FIFO entry?
1440
Yes
No
Wait
1442
Allocate LH tag FIFO entry
1444
Route to AB out
1446
To RH Addr Flow AB in
1448
To FLM Addr Flow AB in
1449
Free LH token
1450
Route addr to snoopers
1452
Snoopers generate Presp
1454
Add Presp to LH Presp FIFO & set F/N flag
1456
End
1458

FIGURE 14B

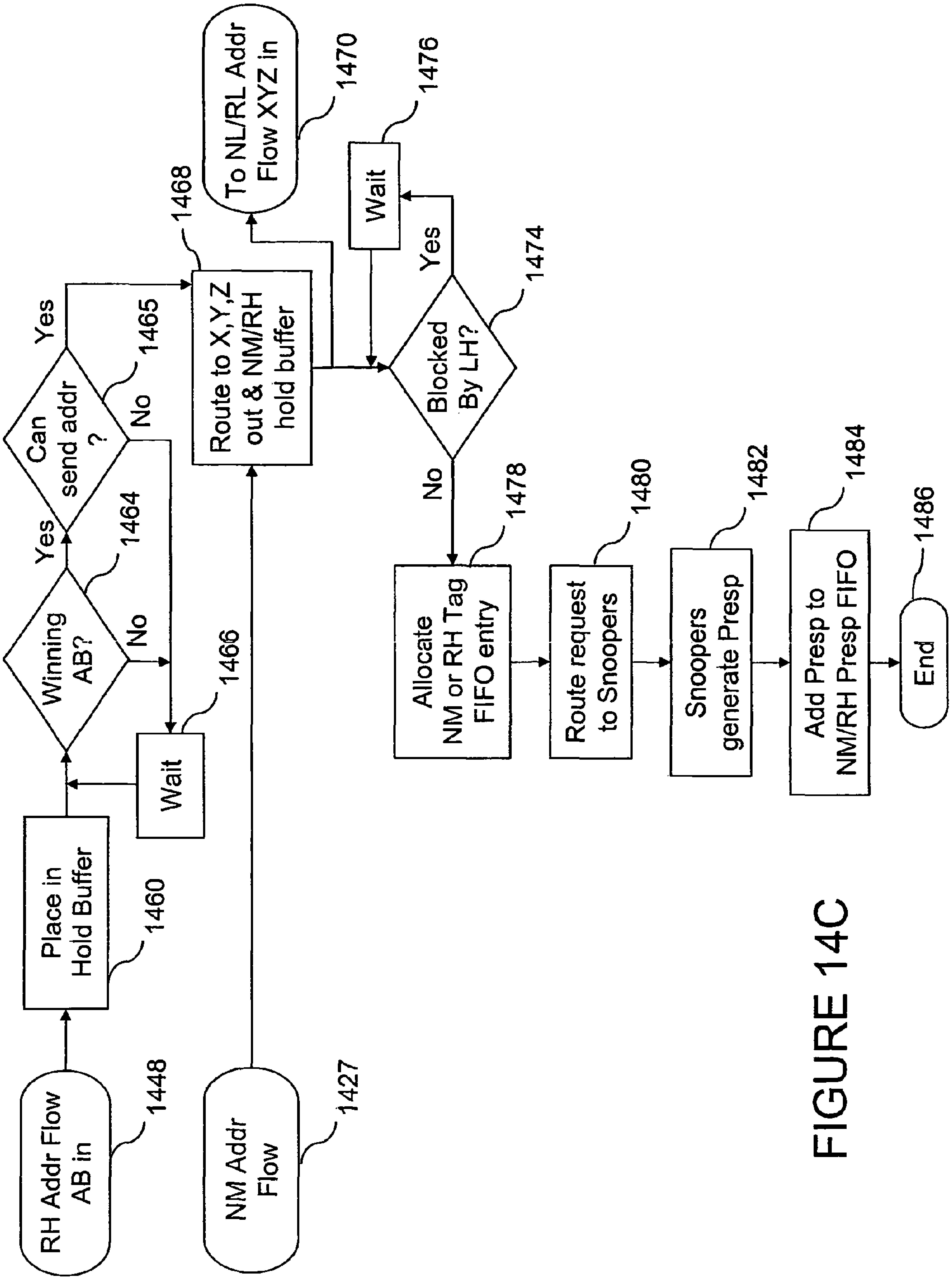

RH Addr Flow AB in
1448
Place in Hold Buffer
1460
Winning AB?
1464
Yes
No
Wait
1466
Can send addr ?
1465
Yes
No
NM Addr Flow
1427
Route to X,Y,Z out & NM/RH hold buffer
1468
To NL/RL Addr Flow XYZ in
1470
Blocked By LH?
1474
Yes
No
Wait
1476
Allocate NM or RH Tag FIFO entry
1478
Route request to Snoopers
1480
Snoopers generate Presp
1482
Add Presp to NM/RH Presp FIFO
1484
End
1486

FIGURE 14C

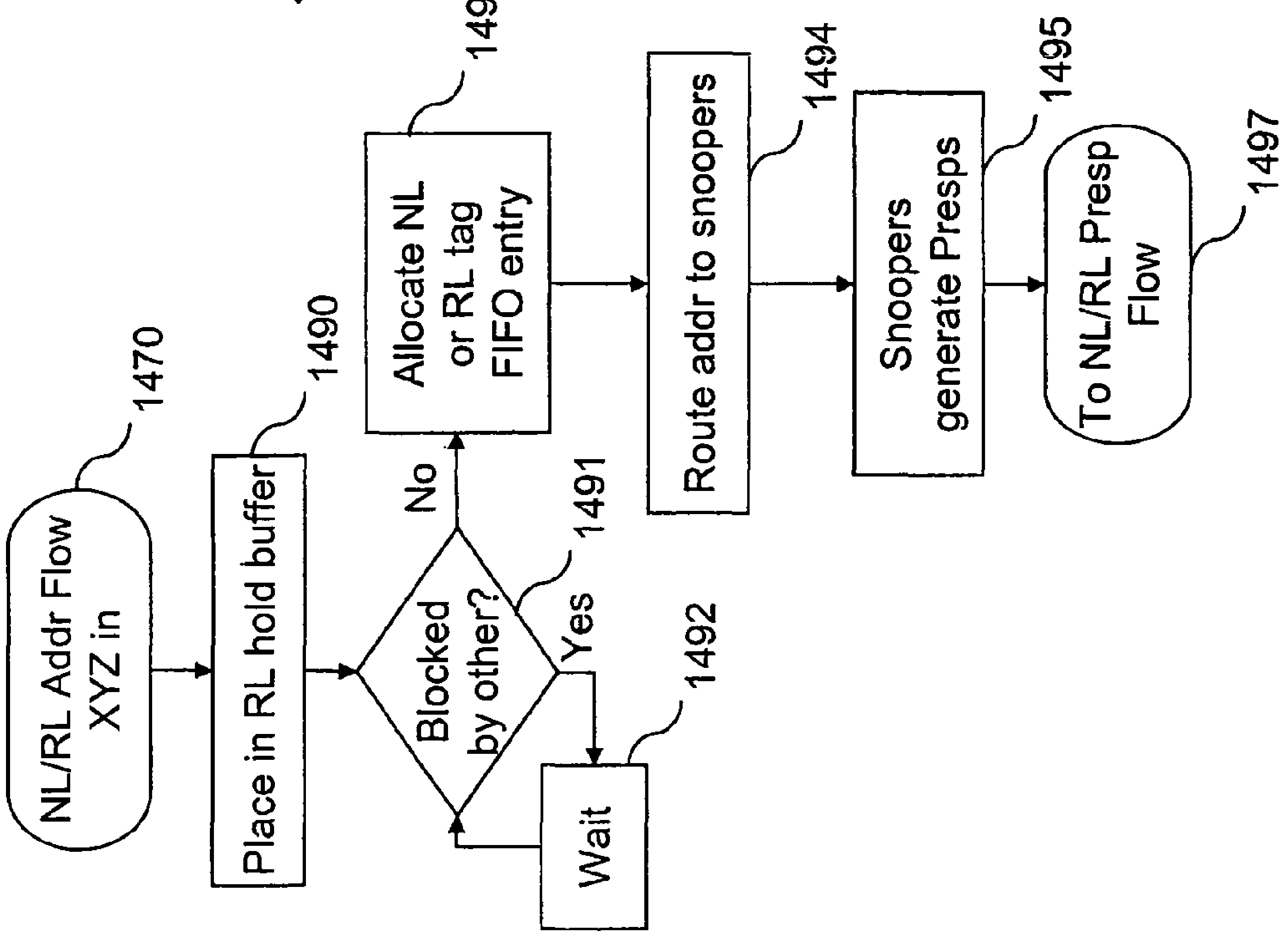


FIGURE 14D

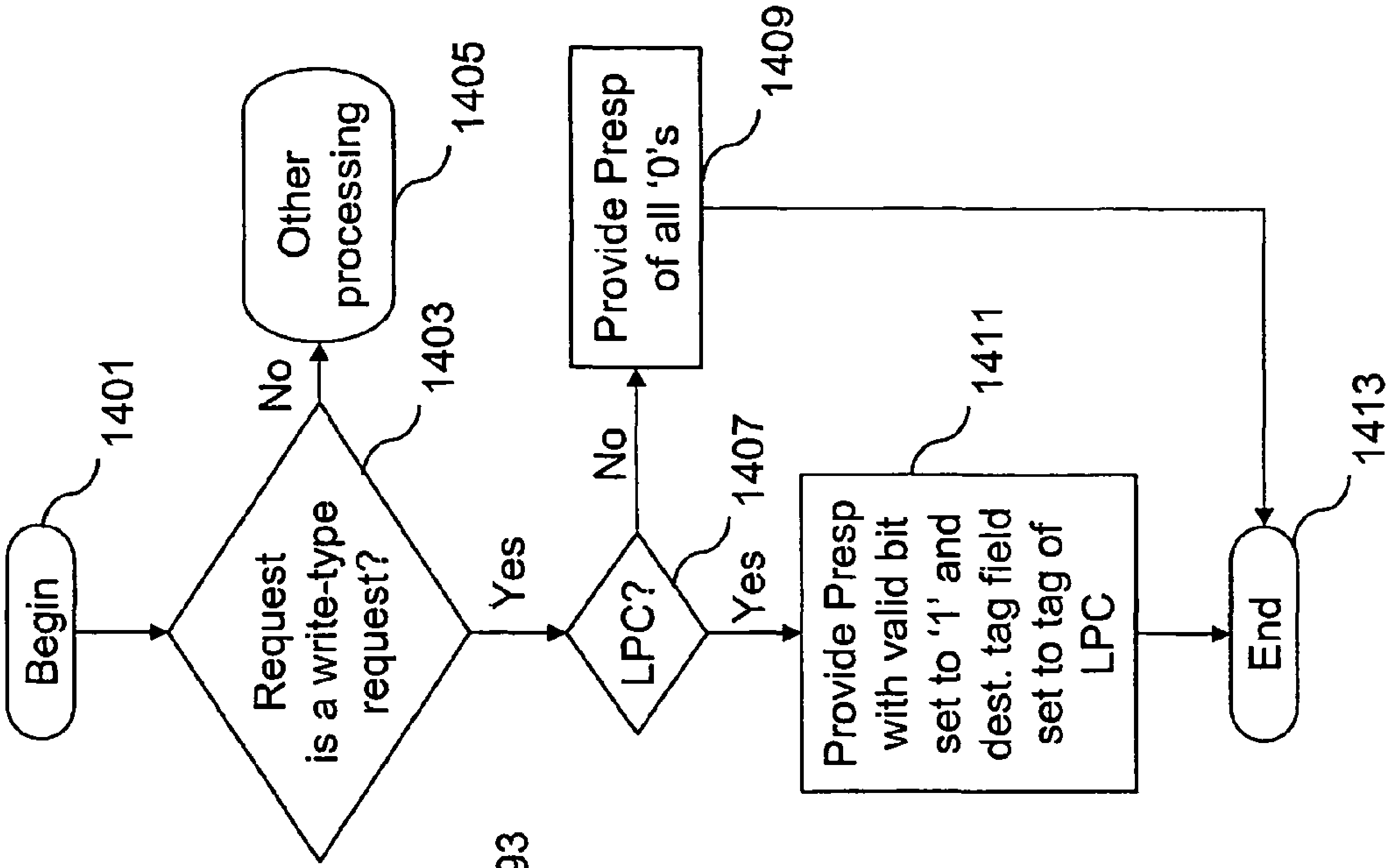


FIGURE 14G

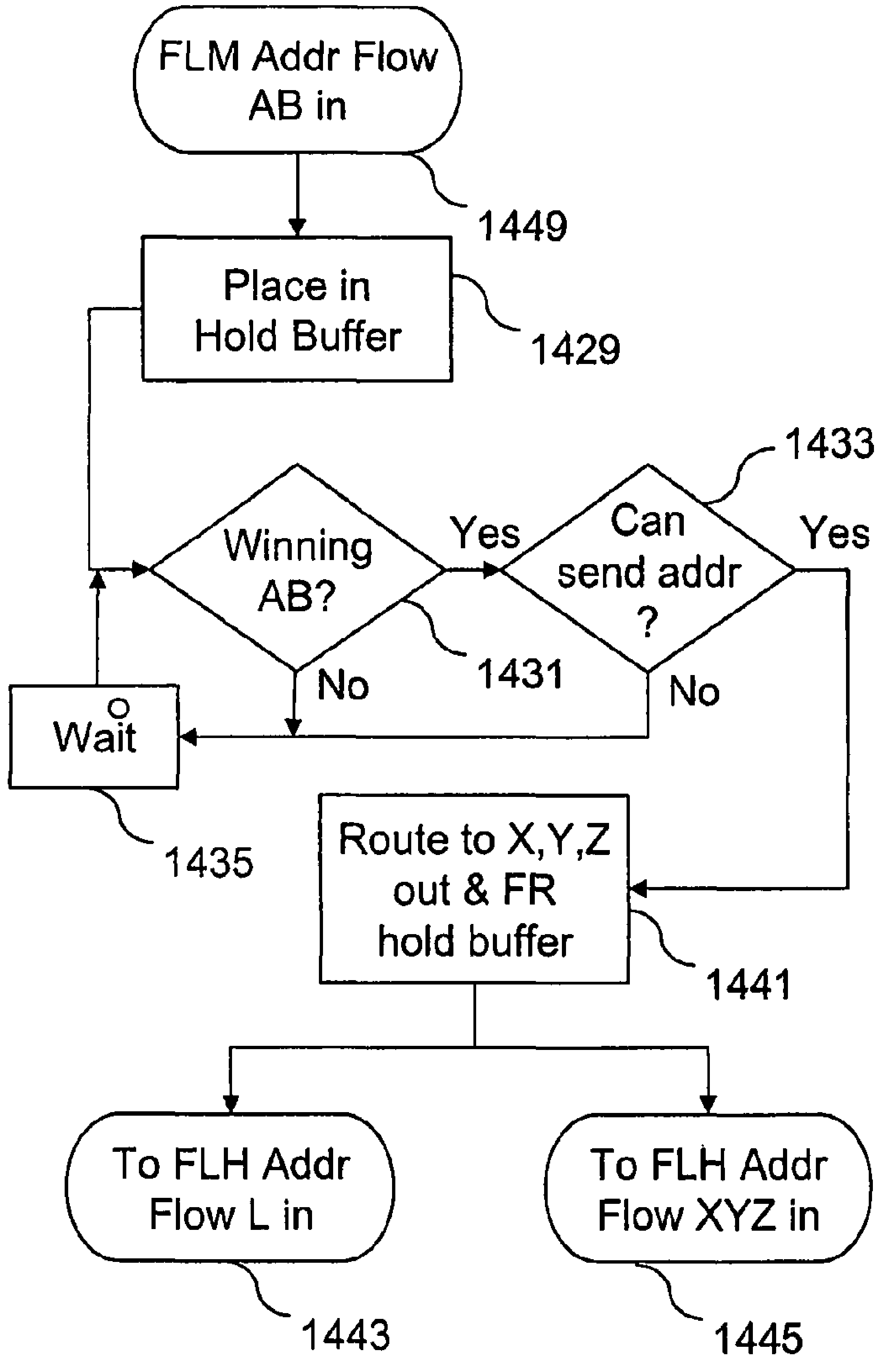


FIGURE 14E

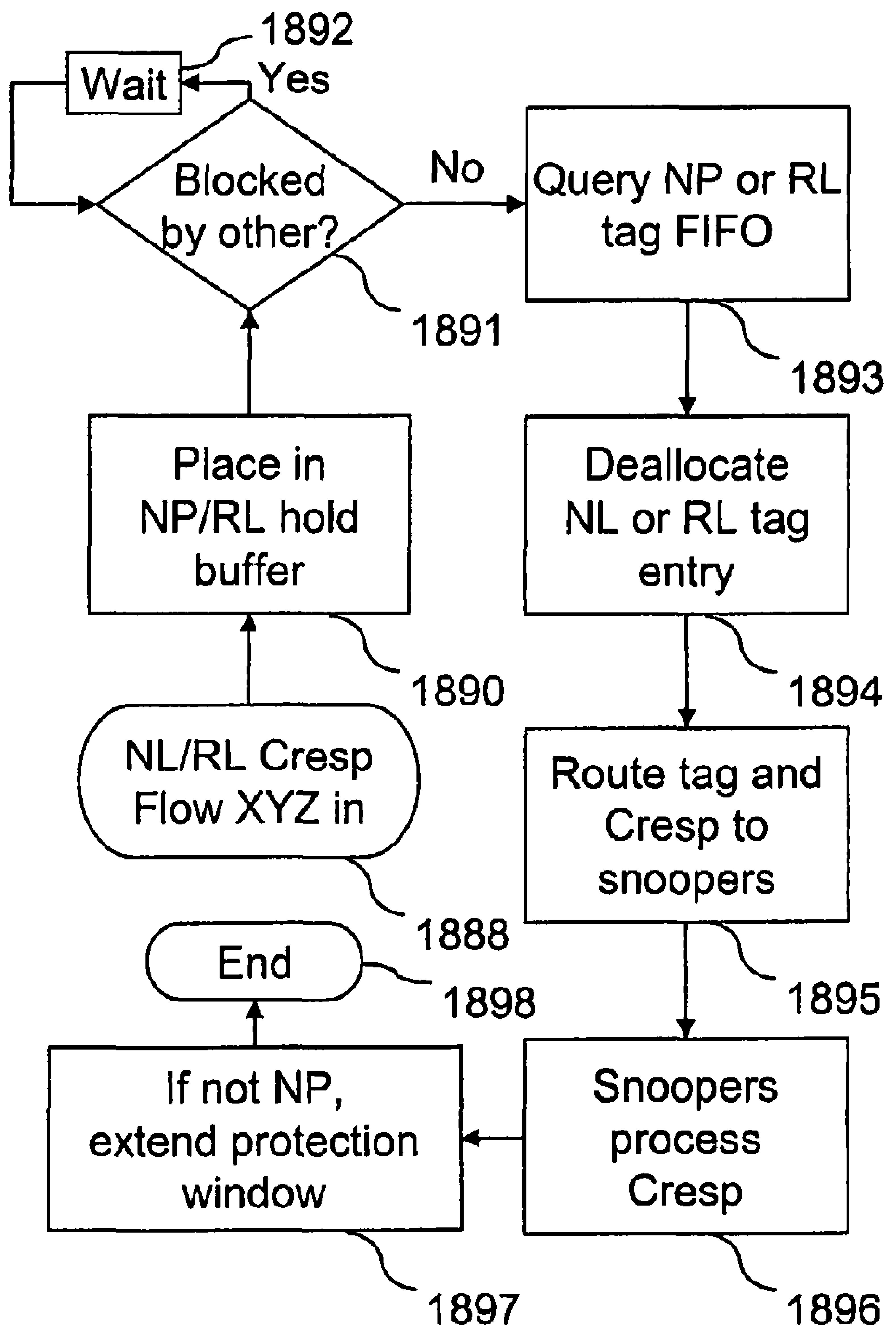


FIGURE 18C

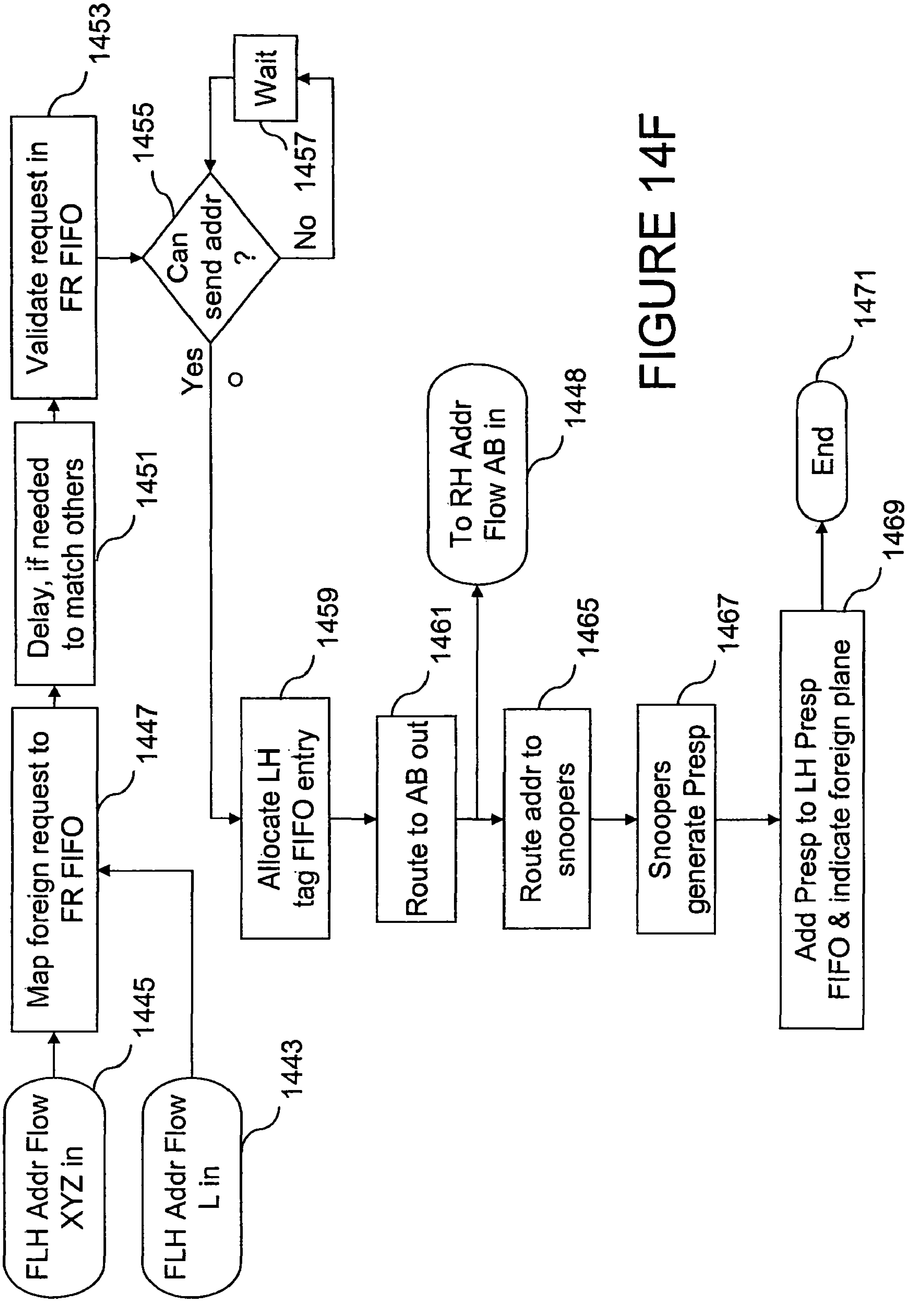

FLH Addr Flow XYZ in
1445
FLH Addr Flow L in
1443
Map foreign request to FR FIFO
1447
Delay, if needed to match others
1451
Validate request in FR FIFO
1453
Can send addr ?
1455
Yes
No
Wait
1457
Allocate LH tag FIFO entry
1459
Route to AB out
1461
To RH Addr Flow AB in
1448
Route addr to snoopers
1465
Snoopers generate Presp
1467
Add Presp to LH Presp FIFO & indicate foreign plane
1469
End
1471

FIGURE 14F

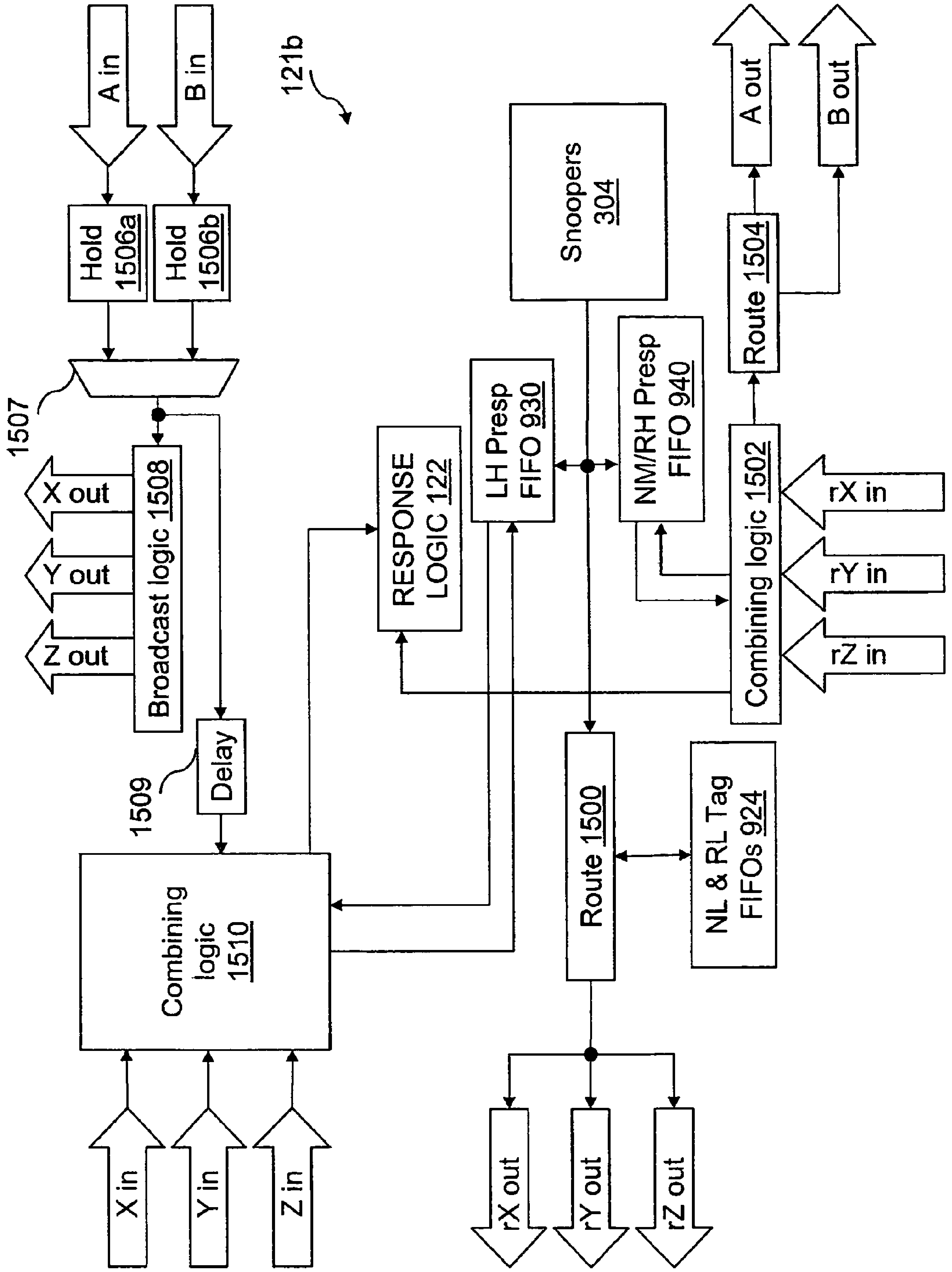

121b
A in
B in
Hold 1506a
Hold 1506b
1507
Broadcast logic 1508
X out
Y out
Z out
Delay
1509
Combining logic 1510
X in
Y in
Z in
RESPONSE LOGIC 122
LH Presp FIFO 930
Snoopers 304
NM/RH Presp FIFO 940
Route 1500
NL & RL Tag FIFOs 924
rX out
rY out
rZ out
Route 1504
A out
B out
Combining logic 1502
rX in
rY in
rZ in

FIGURE 15

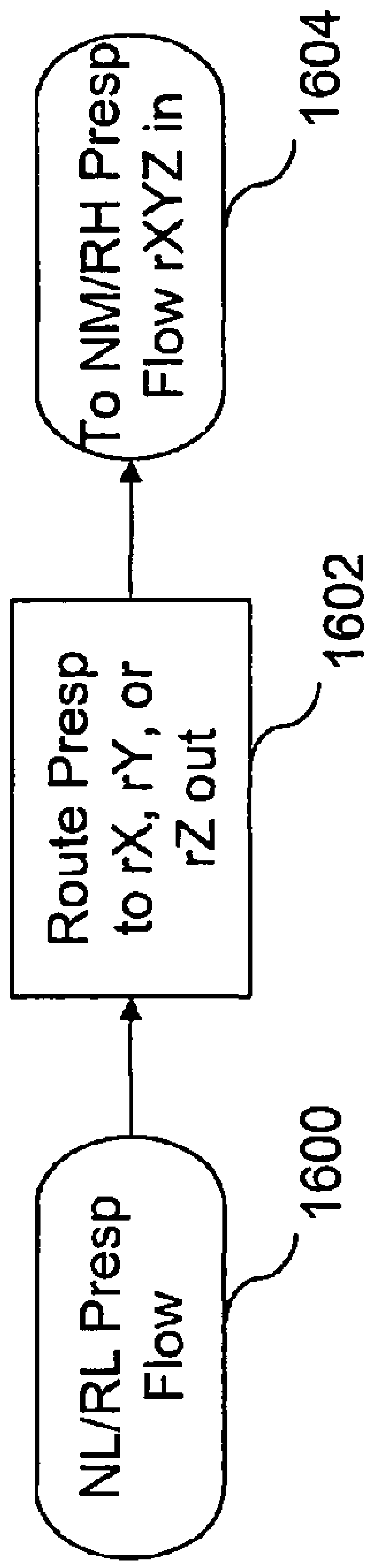


FIGURE 16A

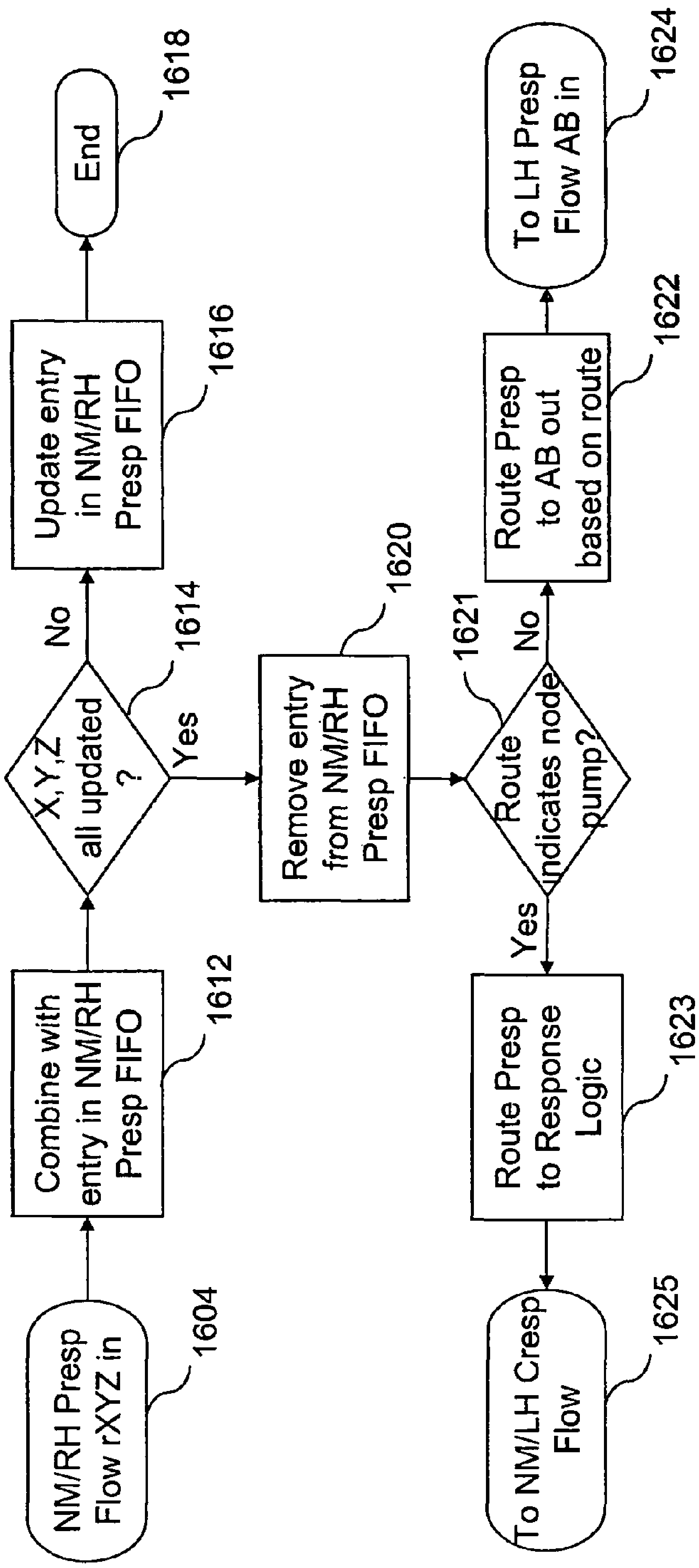


FIGURE 16B

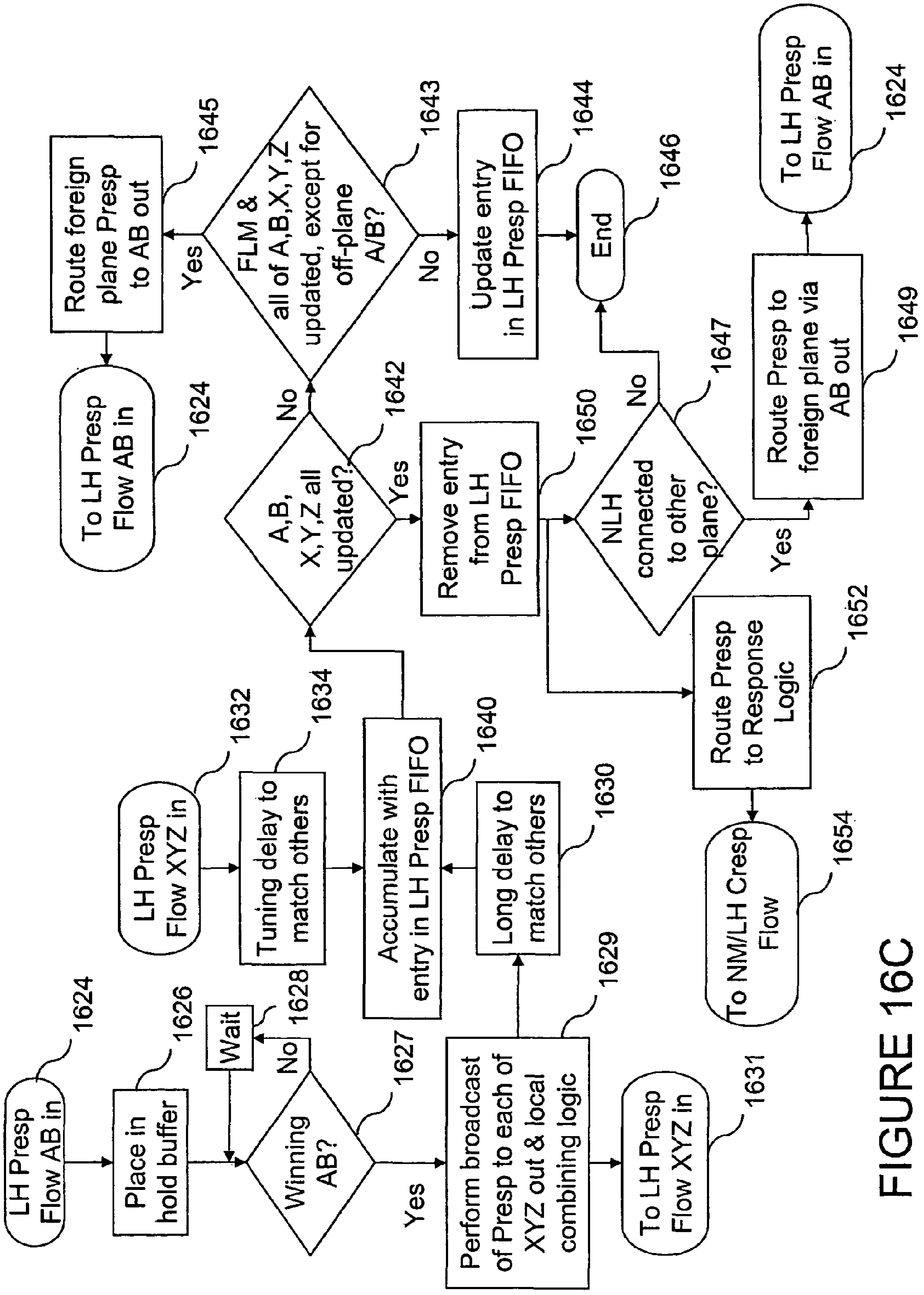

LH Presp Flow AB in
1624
Place in hold buffer
1626
Wait
1628
Winning AB?
1627
No
Yes
Perform broadcast of Presp to each of XYZ out & local combining logic
1629
To LH Presp Flow XYZ in
1631
LH Presp Flow XYZ in
1632
Tuning delay to match others
1634
Accumulate with entry in LH Presp FIFO
1640
Long delay to match others
1630
A,B, X,Y,Z all updated?
1642
No
Yes
FLM & all of A,B,X,Y,Z updated, except for off-plane A/B?
1643
Yes
No
Route foreign plane Presp to AB out
1645
To LH Presp Flow AB in
1624
Update entry in LH Presp FIFO
1644
End
1646
Remove entry from LH Presp FIFO
1650
NLH connected to other plane?
1647
No
Yes
Route Presp to foreign plane via AB out
1649
To LH Presp Flow AB in
1624
Route Presp to Response Logic
1652
To NM/LH Cresp Flow
1654

FIGURE 16C

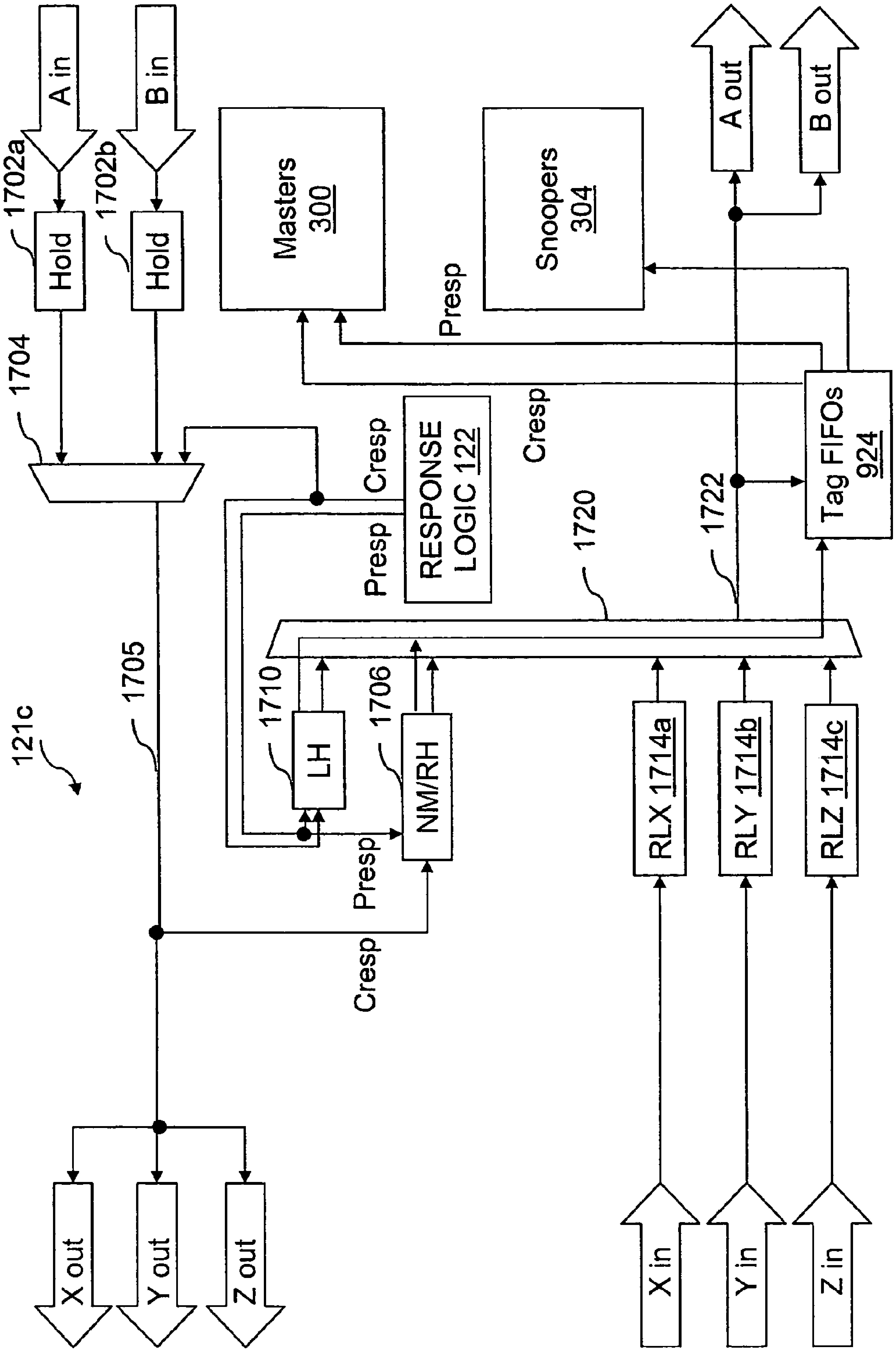

A in
B in
1702a
1702b
Hold
Hold
Masters
300
Snoopers
304
A out
B out
Presp
1704
Cresp
RESPONSE LOGIC 122
Cresp
Presp
1720
1722
Tag FIFOs
924
121c
1705
1710
1706
LH
NM/RH
Presp
Cresp
RLX 1714a
RLY 1714b
RLZ 1714c
X out
Y out
Z out
X in
Y in
Z in

FIGURE 17

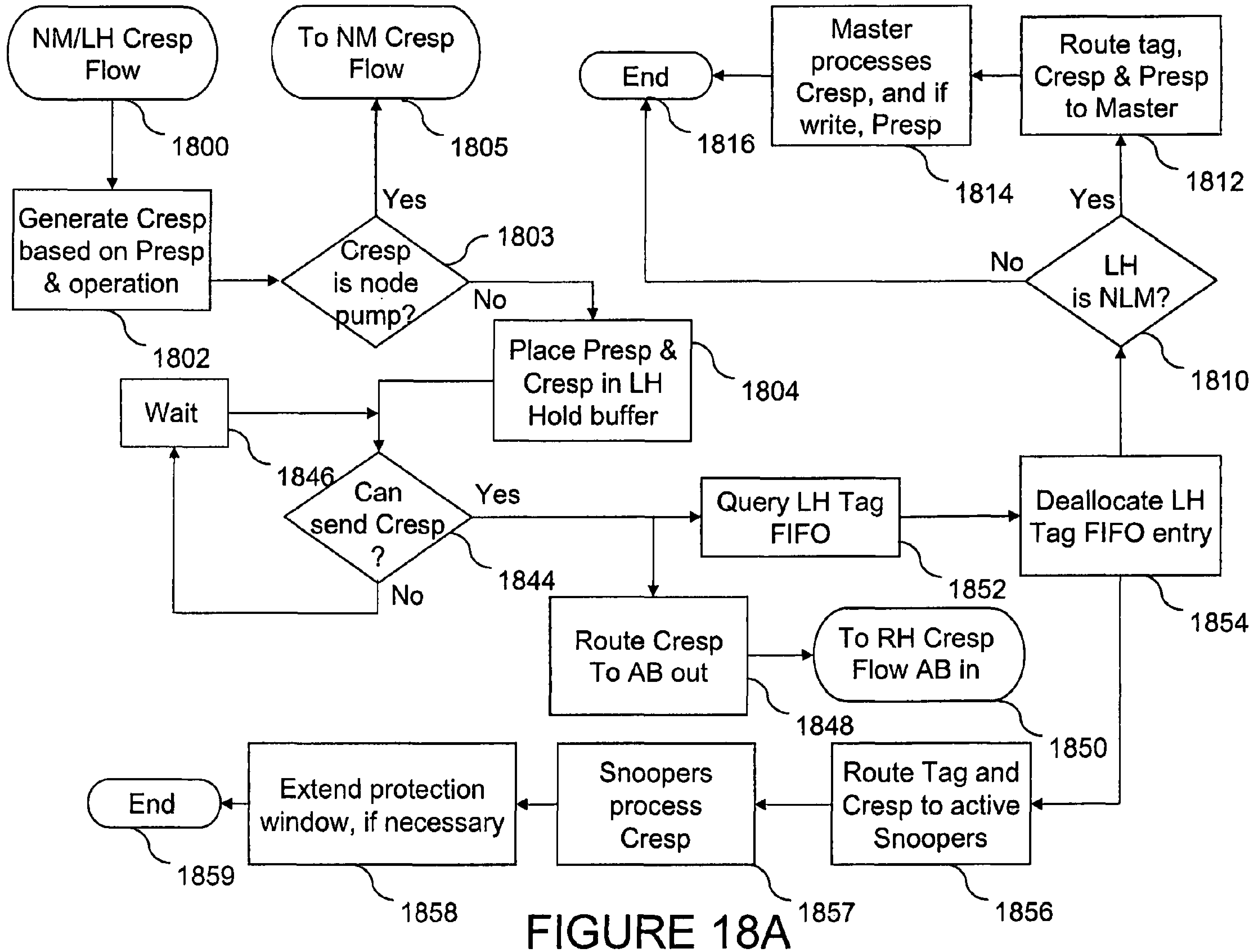

NM/LH Cresp Flow
1800
Generate Cresp based on Presp & operation
1802
Cresp is node pump?
1803
Yes
To NM Cresp Flow
1805
No
Place Presp & Cresp in LH Hold buffer
1804
Wait
1846
Can send Cresp ?
1844
Yes
No
Query LH Tag FIFO
1852
Route Cresp To AB out
1848
To RH Cresp Flow AB in
1850
Deallocate LH Tag FIFO entry
1854
LH is NLM?
1810
Yes
No
Route tag, Cresp & Presp to Master
1812
Master processes Cresp, and if write, Presp
1814
End
1816
Route Tag and Cresp to active Snoopers
1856
Snoopers process Cresp
1857
Extend protection window, if necessary
1858
End
1859

FIGURE 18A

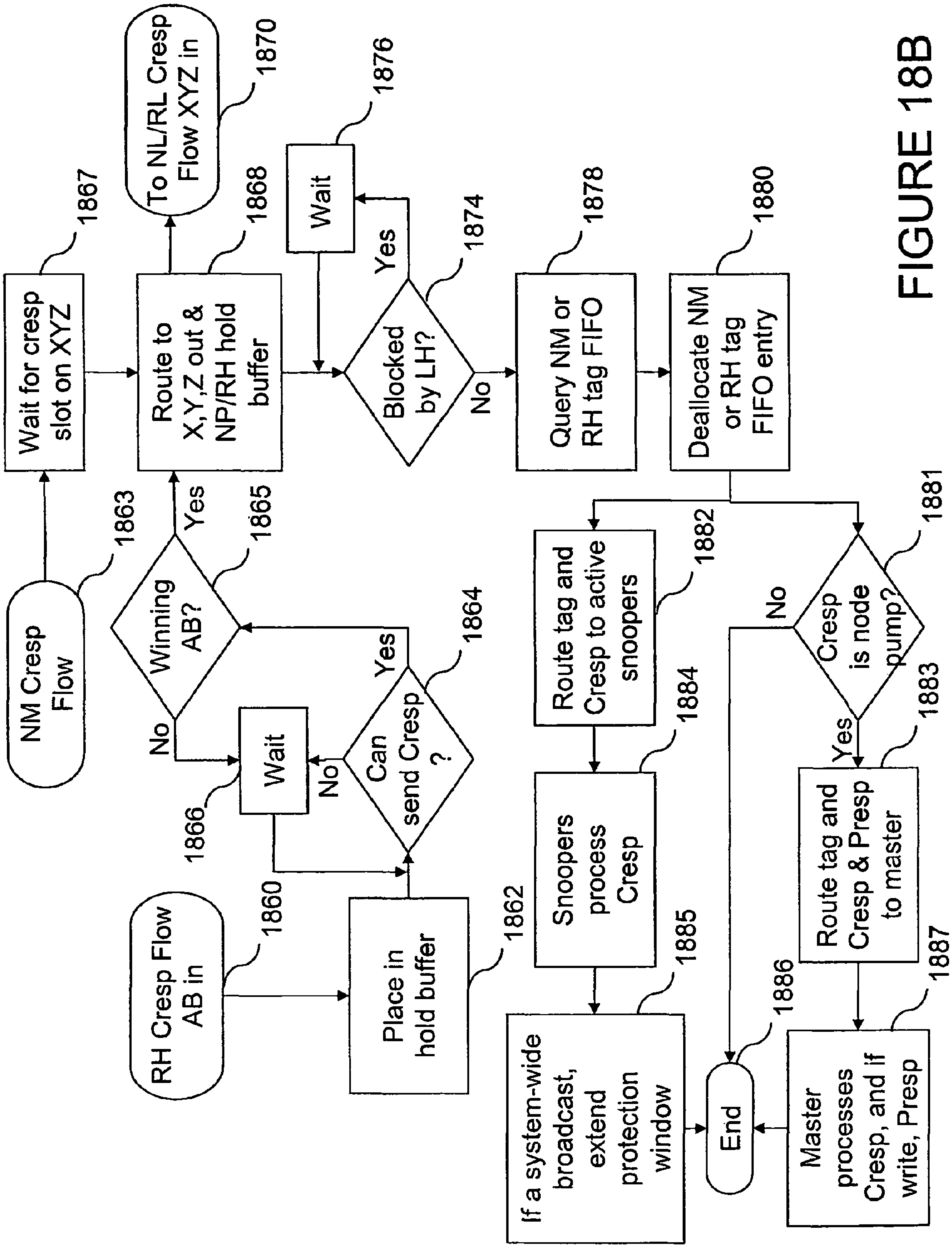

NM Cresp Flow
1863
Wait for cresp slot on XYZ
1867
Route to X,Y,Z out & NP/RH hold buffer
1868
To NL/RL Cresp Flow XYZ in
1870
Wait
1876
Blocked by LH?
1874
Yes
No
Query NM or RH tag FIFO
1878
Deallocate NM or RH tag FIFO entry
1880
Winning AB?
1865
Yes
No
Wait
1866
Can send Cresp ?
1864
Yes
No
RH Cresp Flow AB in
1860
Place in hold buffer
1862
Route tag and Cresp to active snoopers
1882
Snoopers process Cresp
1884
If a system-wide broadcast, extend protection window
1885
Cresp is node pump?
1881
No
Yes
Route tag and Cresp & Presp to master
1883
End
1886
Master processes Cresp, and if write, Presp
1887

FIGURE 18B

DATA PROCESSING SYSTEM, METHOD AND INTERCONNECT FABRIC SUPPORTING MULTIPLE PLANES OF PROCESSING NODES

CROSS-REFERENCE TO RELATED APPLICATION(S)

The present application is related to the following U.S. Patent Application(s), which are assigned to the assignee hereof and incorporated herein by reference in their entireties:

U.S. patent application Ser. No. 11/055,305; and

U.S. patent application Ser. No. 11/054,820.

BACKGROUND OF THE INVENTION

1. Technical Field

The present invention relates in general to data processing systems and, in particular, to an improved interconnect fabric for data processing systems.

2. Description of the Related Art

A conventional symmetric multiprocessor (SMP) computer system, such as a server computer system, includes multiple processing units all coupled to a system interconnect, which typically comprises one or more address, data and control buses. Coupled to the system interconnect is a system memory, which represents the lowest level of volatile memory in the multiprocessor computer system and which generally is accessible for read and write access by all processing units. In order to reduce access latency to instructions and data residing in the system memory, each processing unit is typically further supported by a respective multi-level cache hierarchy, the lower level(s) of which may be shared by one or more processor cores.

SUMMARY OF THE INVENTION

As the clock frequencies at which processing units are capable of operating have risen and system scales have increased, the latency of communication between processing units via the system interconnect has become a critical performance concern. To address this performance concern, various interconnect designs have been proposed and/or implemented that are intended to improve performance and scalability over conventional bused interconnects.

The present invention provides an improved data processing system, interconnect fabric and method of communication in a data processing system. In one embodiment, a data processing system includes a first plane including a first plurality of processing nodes, each including multiple processing units, and a second plane including a second plurality of processing nodes, each including multiple processing units. The data processing system also includes a plurality of point-to-point first tier links. Each of the first plurality and second plurality of processing nodes includes one or more first tier links among the plurality of first tier links, where the first tier link(s) within each processing node connect a pair of processing units in the same processing node for communication. The data processing system further includes a plurality of point-to-point second tier links. At least a first of the plurality of second tier links connects processing units in different ones of the first plurality of processing nodes, at least a second of the plurality of second tier links connects processing units in different ones of the second plurality of processing nodes, and at least a third of the plurality of second tier links connects a processing unit in the first plane to a processing unit in the second plane.

All objects, features, and advantages of the present invention will become apparent in the following detailed written description.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. However, the invention, as well as a preferred mode of use, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

FIG. **1** is a high level block diagram of a processing unit in accordance with the present invention;

FIGS. **2**A-**2**B together depict a high level block diagram of an exemplary data processing system in accordance with the present invention;

FIG. **3** is a time-space diagram of an exemplary operation including a request phase, a partial response phase and a combined response phase;

FIGS. **4**A and **4**B respectively depict time-space diagrams of exemplary operations of system-wide scope and node-only scope within the data processing system of FIG. **2**A-**2**B;

FIGS. **5**A-**5**D illustrate the information flow of the exemplary operation depicted in FIG. **4**A;

FIGS. **6**A-**6**B depict an exemplary data flow for an exemplary system-wide broadcast operation in accordance with the present invention;

FIGS. **7**A-**7**B illustrate a first exemplary link information allocation for the first tier links and intra-plane second tier links in accordance with the present invention;

FIG. **7**C depicts an exemplary link information allocation for inter-plane second tier links in accordance with the present invention;

FIG. **8** is an exemplary embodiment of a partial response field for a write request that is included within the link information allocation;

FIG. **9** is a block diagram illustrating a portion of the interconnect logic of FIG. **1** utilized in the request phase of an operation;

FIG. **10** is a more detailed block diagram of the local hub address launch buffer of FIG. **9**;

FIG. **11** is a more detailed block diagram of the tag FIFO queues of FIG. **9**;

FIGS. **12**A and **12**B are more detailed block diagrams of the local hub partial response FIFO queue and remote hub partial response FIFO queue of FIG. **9**, respectively;

FIG. **13** is a time-space diagram illustrating the tenures of a system-wide broadcast operation with respect to the data structures depicted in FIG. **9**;

FIG. **14**A-**14**F are flowcharts respectively depicting the request phase of an operation at a local master, local hub, remote hub, remote leaf, foreign local master and foreign local hub in accordance with the present invention;

FIG. **14**G is a high level logical flowchart of an exemplary method of generating a partial response at a snooper in accordance with the present invention;

FIG. **15** is a block diagram illustrating a portion of the interconnect logic of FIG. **1** utilized in the partial response phase of an operation;

FIG. **16**A-**16**C are flowcharts respectively depicting the partial response phase of an operation at a remote leaf, remote hub, local hub, and local master;

FIG. **17** is a block diagram illustrating a portion of the interconnect logic of FIG. **1** utilized in the combined response phase of an operation;

FIG. **18**A-**18**C are flowcharts respectively depicting the combined response phase of an operation at a local hub, remote hub, and remote leaf; and

FIG. **19** is a more detailed block diagram of an exemplary snooping component of the data processing system of FIGS. **2**A-**2**B.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENT

I. Processing Unit and Data Processing System

With reference now to the figures and, in particular, with reference to FIG. **1**, there is illustrated a high level block diagram of an exemplary embodiment of a processing unit **100** in accordance with the present invention. In the depicted embodiment, processing unit **100** is a single integrated circuit including two processor cores **102***a*, **102***b* for independently processing instructions and data. Each processor core **102** includes at least an instruction sequencing unit (ISU) **104** for fetching and ordering instructions for execution and one or more execution units **106** for executing instructions. The instructions executed by execution units **106** may include, for example, fixed and floating point arithmetic instructions, logical instructions, and instructions that request read and write access to a memory block.

The operation of each processor core **102***a*, **102***b* is supported by a multi-level volatile memory hierarchy having at its lowest level one or more shared system memories **132** (only one of which is shown in FIG. **1**) and, at its upper levels, one or more levels of cache memory. As depicted, processing unit **100** includes an integrated memory controller (IMC) **124** that controls read and write access to a system memory **132** in response to requests received from processor cores **102***a*, **102***b* and operations snooped on an interconnect fabric (described below) by snoopers **126**.

In the illustrative embodiment, the cache memory hierarchy of processing unit **100** includes a store-through level one (L1) cache **108** within each processor core **102***a*, **102***b* and a level two (L2) cache **110** shared by all processor cores **102***a*, **102***b* of the processing unit **100**. L2 cache **110** includes an L2 array and directory **114**, masters **112** and snoopers **116**. Masters **112** initiate transactions on the interconnect fabric and access L2 array and directory **114** in response to memory access (and other) requests received from the associated processor cores **102***a*, **102***b*. Snoopers **116** detect operations on the interconnect fabric, provide appropriate responses, and perform any accesses to L2 array and directory **114** required by the operations. Although the illustrated cache hierarchy includes only two levels of cache, those skilled in the art will appreciate that alternative embodiments may include additional levels (L3, L4, etc.) of on-chip or off-chip in-line or look aside cache, which may be fully inclusive, partially inclusive, or non-inclusive of the contents the upper levels of cache.

As further shown in FIG. **1**, processing unit **100** includes integrated interconnect logic **120** by which processing unit **100** may be coupled to the interconnect fabric as part of a larger data processing system. In the depicted embodiment, interconnect logic **120** supports an arbitrary number t**1** of "first tier" interconnect links, which in this case include in-bound and out-bound X, Y and Z link pairs. Interconnect logic **120** further supports an arbitrary number t**2** of second tier link pairs, designated in FIG. **1** as in-bound and out-bound A and B links. With these first and second tier links, each processing unit **100** may be coupled for bi-directional communication to up to t**1**/**2**+t**2**/**2** (in this case, five) other processing units **100**. Interconnect logic **120** includes request logic **121***a*, partial response logic **121***b*, combined response logic **121***c* and data logic **121***d* for processing and forwarding information during different phases of operations. In addition, interconnect logic **120** includes a configuration register **123** including a plurality of mode bits utilized to configure processing unit **100**. As further described below, these mode bits preferably include: (1) a first set of one or more mode bits that selects a desired link information allocation for the first and second tier links; (2) a second set of mode bits that specify which of the first and second tier links of the processing unit **100** are connected to other processing units **100**; (3) a third set of mode bits that determines a programmable duration of a protection window extension; and (4) a fourth set of mode bits that predictively selects a scope of broadcast for operations initiated by the processing unit **100** on an operation-by-operation basis from among a node-only broadcast scope or a system-wide scope, as described in above-referenced U.S. patent application Ser. No. 11/055,305.

Each processing unit **100** further includes an instance of response logic **122**, which implements a portion of a distributed coherency signaling mechanism that maintains cache coherency between the cache hierarchy of processing unit **100** and those of other processing units **100**. Finally, each processing unit **100** includes an integrated I/O (input/output) controller **128** supporting the attachment of one or more I/O devices, such as I/O device **130**. I/O controller **128** may issue operations and receive data on the X, Y, Z, A and B links in response to requests by I/O device **130**.

Referring now to FIGS. **2**A-**2**B, there is depicted a block diagram of an exemplary embodiment of a data processing system **200** formed of multiple processing units **100** in accordance with the present invention. As shown, data processing system **200** includes 16 processing nodes **202***aap***1**-**202***dap***1**, **202***abp***1**-**202***dbp***1**, **202***aap***2**-**202***dap***2**, and **202***abp***2**-**202***dbp***2**, arranged in two planes (p**1** and p**2**) each containing 8 processing nodes **202**. Each plane of 8 processing nodes in turn contains two sets (a and b) of 4 processing nodes, and each of the processing nodes **202** can be designated as a, b, c or d to indicate its relative position in its set. Thus, the nomenclature of the reference numerals associated with each processing node **202** includes 3 components indicative of the location of the processing node **202** within data processing system **200**: a set position component, set component, and plane component. For example, processing node **202***aap***1** refers to the processing node **202** in set position "a" within set "a" of plane **1**, while processing node **202***dbp***2** refers to the processing node **202** in set position "d" within set "b" of plane **2**.

In the depicted embodiment, each processing node **202** is realized as a multi-chip module (MCM) comprising a package containing four processing units **100***a*-**100***d*. The processing units **100** within each processing node **202** are coupled for point-to-point communication by the processing units' first tier X, Y, and Z links, as shown. Each processing unit **100** may be further coupled to processing units **100** in two different processing nodes **202** for point-to-point communication by the processing units' second tier A and B links. In the depicted embodiment, seven of the second tier links in each processing node **202** are coupled to processing units **100** in other processing nodes **202** in a same plane, and one of the second tier links is coupled to a processing unit **100** in a processing node **202** in a different plane. For example, the A link of processing unit **100***a* of processing node **202***aap***1** is coupled to the A link of processing unit **100***a* of processing node **202***aap***2**. Although illustrated in FIGS. **2**A-**2**B with a double-headed arrow, it should be understood that each pair of X, Y, Z, A and B links are preferably (but not necessarily) implemented as two uni-directional links, rather than as a bi-directional link.

General expressions for forming the topology shown in FIGS. **2**A-**2**B can be given as follows (using the reference numeral nomenclature for processing nodes **202** set forth above):

Node[I][K][L].chip[J].link[K] connects to
Node[J][K][L].chip[I].link[K], for all I≠J; and
Node[I][K][L].chip[I].link[not K] connects to
Node[I][not K][L].chip[I].link[K]; and
Node[I][K][L].chip[I].link[K] connects to
Node[I][K][not L].chip[I].link[K],

where I and J belong to the set {a, b, c, d} and K belongs to the set {A,B}, and L belongs to the set {p1, p2}.

Of course, alternative expressions can be defined to form other functionally equivalent topologies. Moreover, it should be appreciated that the depicted topology is representative but not exhaustive of data processing system topologies embodying the present invention and that other topologies are possible. In such alternative topologies, for example, the number of first tier and second tier links coupled to each processing unit **100** can be an arbitrary number, and the number of processing nodes **202** within each set need not equal the number of processing units **100** per processing node **100**.

Even though fully connected in the manner shown in FIGS. **2**A-**2**B, all processing nodes **202** need not communicate each operation to all other processing nodes **202**. In particular, as noted above, processing units **100** may broadcast operations with a scope limited to their processing node **202** or with a larger scope, such as a system-wide scope including all processing nodes **202**.

As shown in FIG. **19**, an exemplary snooping device **1900** within data processing system **200**, for example, snoopers **116** of L2 (or lower level) cache or snoopers **126** of an IMC **124**, may include one or more base address registers (BARs) **1902** identifying one or more regions of the real address space containing real addresses for which the snooping device **1900** is responsible. Snooping device **1900** may optionally further include hash logic **1904** that performs a hash function on real addresses falling within the region(s) of real address space identified by BAR **1902** to further qualify whether or not the snooping device **1900** is responsible for the addresses. Finally, snooping device **1900** includes a number of snoopers **1906***a*-**1906***m* that access resource **1910** (e.g., L2 cache array and directory **114** or system memory **132**) in response to snooped requests specifying request addresses qualified by BAR **1902** and hash logic **1904**.

As shown, resource **1910** may have a banked structure including multiple banks **1912***a*-**1912***n* each associated with a respective set of real addresses. As is known to those skilled in the art, such banked designs are often employed to support a higher arrival rate of requests for resource **1910** by effectively subdividing resource **1910** into multiple independently accessible resources. In this manner, even if the operating frequency of snooping device **1900** and/or resource **1910** are such that snooping device **1900** cannot service requests to access resource **1910** as fast as the maximum arrival rate of such requests, snooping device **1900** can service such requests without retry as long as the number of requests received for any bank **1912** within a given time interval does not exceed the number of requests that can be serviced by that bank **1912** within that time interval.

Those skilled in the art will appreciate that SMP data processing system **100** can include many additional unillustrated components, such as interconnect bridges, non-volatile storage, ports for connection to networks or attached devices, etc. Because such additional components are not necessary for an understanding of the present invention, they are not illustrated in FIG. **2** or discussed further herein.

II. Exemplary Operation

Referring now to FIG. **3**, there is depicted a time-space diagram of an exemplary operation on the interconnect fabric of data processing system **200** of FIGS. **2**A-**2**B. The operation begins when a master **300** (e.g., a master **112** of an L2 cache **110** or a master within an I/O controller **128**) issues a request **302** on the interconnect fabric. Request **302** preferably includes at least a transaction type indicating a type of desired access and a resource identifier (e.g., real address) indicating a resource to be accessed by the request. Common types of requests preferably include those set forth below in Table I.

TABLE I

Request	Description
READ	Requests a copy of the image of a memory block for query purposes
RWITM (Read-With-Intent-To-Modify)	Requests a unique copy of the image of a memory block with the intent to update (modify) it and requires destruction of other copies, if any
DCLAIM (Data Claim)	Requests authority to promote an existing query-only copy of memory block to a unique copy with the intent to update (modify) it and requires destruction of other copies, if any
DCBZ (Data Cache Block Zero)	Requests authority to create a new unique copy of a memory block without regard to its present state and subsequently modify its contents; requires destruction of other copies, if any
CASTOUT	Copies the image of a memory block from a higher level of memory to a lower level of memory in preparation for the destruction of the higher level copy
WRITE	Requests authority to create a new unique copy of a memory block without regard to its present state and immediately copy the image of the memory block from a higher level memory to a lower level memory in preparation for the destruction of the higher level copy
PARTIAL WRITE	Requests authority to create a new unique copy of a partial memory block without regard to its present state and immediately copy the image of the partial memory block from a higher level memory to a lower level memory in preparation for the destruction of the higher level copy

Further details regarding these operations and an exemplary cache coherency protocol that facilitates efficient handling of these operations may be found in the copending U.S. patent application Ser. No. 11/055,305 incorporated by reference above.

Request **302** is received by snoopers **304**, for example, snoopers **116** of L2 caches **110** and snoopers **126** of IMCs **124**, distributed throughout data processing system **200**. In general, with some exceptions, snoopers **116** in the same L2 cache **110** as the master **112** of request **302** do not snoop request **302** (i.e., there is generally no self-snooping) because a request **302** is transmitted on the interconnect fabric only if the request **302** cannot be serviced internally by a processing unit **100**. Snoopers **304** that receive and process requests **302** each provide a respective partial response **306** representing the response of at least that snooper **304** to request **302**. A snooper **126** within an IMC **124** determines the partial response **306** to provide based, for example, upon whether the snooper **126** is responsible for the request address and whether it has resources available to service the request. A snooper **116** of an L2 cache **110** may determine its partial response **306** based on, for example, the availability of its L2 cache directory **114**, the availability of a snoop logic instance within snooper **116** to handle the request, and the coherency state associated with the request address in L2 cache directory **114**.

The partial responses **306** of snoopers **304** are logically combined either in stages or all at once by one or more instances of response logic **122** to determine a combined response (CR) **310** to request **302**. In one preferred embodiment, which will be assumed hereinafter, the instance of response logic **122** responsible for generating combined response **310** is located in the processing unit **100** containing the master **300** that issued request **302**. Response logic **122** provides combined response **310** to master **300** and snoopers **304** via the interconnect fabric to indicate the response (e.g., success, failure, retry, etc.) to request **302**. If the CR **310** indicates success of request **302**, CR **310** may indicate, for example, a data source for a requested memory block, a cache state in which the requested memory block is to be cached by master **300**, and whether "cleanup" operations invalidating the requested memory block in one or more L2 caches **110** are required.

In response to receipt of combined response **310**, one or more of master **300** and snoopers **304** typically perform one or more operations in order to service request **302**. These operations may include supplying data to master **300**, invalidating or otherwise updating the coherency state of data cached in one or more L2 caches **110**, performing castout operations, writing back data to a system memory **132**, etc. If required by request **302**, a requested or target memory block may be transmitted to or from master **300** before or after the generation of combined response **310** by response logic **122**.

In the following description, the partial response **306** of a snooper **304** to a request **302** and the operations performed by the snooper **304** in response to the request **302** and/or its combined response **310** will be described with reference to whether that snooper is a Highest Point of Coherency (HPC), a Lowest Point of Coherency (LPC), or neither with respect to the request address specified by the request. An LPC is defined herein as a memory device or I/O device that serves as the repository for a memory block. In the absence of a HPC for the memory block, the LPC holds the true image of the memory block and has authority to grant or deny requests to generate an additional cached copy of the memory block. For a typical request in the data processing system embodiment of FIGS. **1** and **2A-2B**, the LPC will be the memory controller **124** for the system memory **132** holding the referenced memory block. An HPC is defined herein as a uniquely identified device that caches a true image of the memory block (which may or may not be consistent with the corresponding memory block at the LPC) and has the authority to grant or deny a request to modify the memory block. Descriptively, the HPC may also provide a copy of the memory block to a requester in response to an operation that does not modify the memory block. Thus, for a typical request in the data processing system embodiment of FIGS. **1** and **2A-2B**, the HPC, if any, will be an L2 cache **110**. Although other indicators may be utilized to designate an HPC for a memory block, a preferred embodiment of the present invention designates the HPC, if any, for a memory block utilizing selected cache coherency state(s) within the L2 cache directory **114** of an L2 cache **110**.

Still referring to FIG. **3**, the HPC, if any, for a memory block referenced in a request **302**, or in the absence of an HPC, the LPC of the memory block, preferably has the responsibility of protecting the transfer of ownership of a memory block, if necessary, in response to a request **302**. In the exemplary scenario shown in FIG. **3**, a snooper **304***n* at the HPC (or in the absence of an HPC, the LPC) for the memory block specified by the request address of request **302** protects the transfer of ownership of the requested memory block to master **300** during a protection window **312***a* that extends from the time that snooper **304***n* determines its partial response **306** until snooper **304***n* receives combined response **310** and during a subsequent window extension **312***b* extending a programmable time beyond receipt by snooper **304***n* of combined response **310**. During protection window **312***a* and window extension **312***b*, snooper **304***n* protects the transfer of ownership by providing partial responses **306** to other requests specifying the same request address that prevent other masters from obtaining ownership (e.g., a retry partial response) until ownership has been successfully transferred to master **300**. Master **300** likewise initiates a protection window **313** to protect its ownership of the memory block requested in request **302** following receipt of combined response **310**.

Because snoopers **304** all have limited resources for handling the CPU and IO requests described above, several different levels of partial responses and corresponding CRs are possible. For example, if a snooper **126** within a memory controller **124** that is responsible for a requested memory block has a queue available to handle a request, the snooper **126** may respond with a partial response indicating that it is able to serve as the LPC for the request. If, on the other hand, the snooper **126** has no queue available to handle the request, the snooper **126** may respond with a partial response indicating that is the LPC for the memory block, but is unable to currently service the request. Similarly, a snooper **116** in an L2 cache **110** may require an available instance of snoop logic and access to L2 cache directory **114** in order to handle a request. Absence of access to either (or both) of these resources results in a partial response (and corresponding CR) signaling an inability to service the request due to absence of a required resource.

III. Timing Considerations

Still referring to FIG. **3**, coherency is maintained during the "handoff" of coherency ownership of a memory block from a snooper **304***n* to a requesting master **300** in the possible presence of other masters competing for ownership of the same memory block through protection window **312***a*, window extension **312***b*, and protection window **313**. Protection window **312***a* and window extension **312***b* must together be of sufficient duration to protect the transfer of coherency ownership of the requested memory block from snooper **304***n* to winning master (WM) **300** in the presence of a competing request **322** by a competing master (CM) **320**. To ensure that protection window **312***a* and window extension **312***b* have sufficient duration to protect the transfer of ownership of the requested memory block from snooper **304***n* to winning master **300**, the latency of communication between processing units **100** is preferably constrained such that the following conditions are met:

$$A\_lat(CM\_S) \leqq A\_lat(CM\_WM) + C\_lat(WM\_S) + \epsilon,$$

where A_lat(CM_S) is the address latency of any competing master (CM) **320** to the snooper (S) **304***n* owning coherence of the requested memory block, A_lat(CM_WM) is the address latency of any competing master (CM) **320** to the "winning" master (WM) **300** that is awarded coherency ownership by snooper **304***n*, C_lat(WM_S) is the combined response latency from the time that the combined response is received by the winning master (WM) **300** to the time the combined response is received by the snooper (S) **304***n* owning the requested memory block, and ϵ is the duration of window extension **312***b*.

If the foregoing timing constraint, which is applicable to a system of arbitrary topology, is not satisfied, the request **322** of the competing master **320** may be received (1) by winning master **300** prior to winning master **300** assuming coherency ownership and initiating protection window **312***b* and (2) by snooper **304***n* after protection window **312***a* and window extension **312***b* end. In such cases, neither winning master **300** nor snooper **304***n* will provide a partial response to competing request **322** that prevents competing master **320** from assuming coherency ownership of the memory block and reading non-coherent data from memory. However, to avoid this coherency error, window extension **312***b* can be programmably set (e.g., by appropriate setting of configuration register **123**) to an arbitrary length (ϵ) to compensate for latency variations or the shortcomings of a physical implementation that may otherwise fail to satisfy the timing constraint that must be satisfied to maintain coherency. Thus, by solving the above equation for ϵ, the ideal length of window extension **312***b* for any implementation can be determined. For the data processing system embodiment of FIGS. **2**A-**2**B, it is preferred if ϵ has a duration equal to the latency of one first tier link chip-hop for broadcast operations having a scope including multiple processing nodes **202** and has a duration of zero for operations of node-only scope.

Several observations may be made regarding the foregoing timing constraint. First, the address latency from the competing master **320** to the owning snooper **304***a* has no necessary lower bound, but must have an upper bound. The upper bound is designed for by determining the worst case latency attainable given, among other things, the maximum possible oscillator drift, the longest links coupling processing units **100**, the maximum number of accumulated stalls, and guaranteed worst case throughput. In order to ensure the upper bound is observed, the interconnect fabric must ensure non-blocking behavior.

Second, the address latency from the competing master **320** to the winning master **300** has no necessary upper bound, but must have a lower bound. The lower bound is determined by the best case latency attainable, given, among other things, the absence of stalls, the shortest possible link between processing units **100** and the slowest oscillator drift given a particular static configuration.

Although for a given operation, each of the winning master **300** and competing master **320** has only one timing bound for its respective request, it will be appreciated that during the course of operation any processing unit **100** may be a winning master for some operations and a competing (and losing) master for other operations. Consequently, each processing unit **100** effectively has an upper bound and a lower bound for its address latency.

Third, the combined response latency from the time that the combined response is generated to the time the combined response is observed by the winning master **300** has no necessary lower bound (the combined response may arrive at the winning master **300** at an arbitrarily early time), but must have an upper bound. By contrast, the combined response latency from the time that a combined response is generated until the combined response is received by the snooper **304***n* has a lower bound, but no necessary upper bound (although one may be arbitrarily imposed to limit the number of operations concurrently in flight).

Fourth, there is no constraint on partial response latency. That is, because all of the terms of the timing constraint enumerated above pertain to request/address latency and combined response latency, the partial response latencies of snoopers **304** and competing master **320** to winning master **300** have no necessary upper or lower bounds.

IV. Broadcast Flow of Exemplary Operations

Referring now to FIG. **4**A, which will be described in conjunction with FIGS. **5**A-**5**D, there is illustrated a time-space diagram of an exemplary operation flow of an operation of system-wide scope in a simplified data processing system **500** comprising two planes each containing two processing nodes **202** of 4 processing units **100**. In these figures, the various processing nodes **202** within data processing system **500** are identified using the same tripartite nomenclature described above. In addition, individual processing units **100** are uniquely identified with a processing node identifier and a positional identifier indicating the relative position of the processing unit **100** within its processing node **202**. Thus, for example, processing unit **100***aap***1***c* refers to processing unit **100***c* of processing node **202***aap***1**. In addition, each processing unit **100** is tagged with a functional identifier indicating its function relative to the other processing units **100** participating in the operation. These functional identifiers include: (1) native local master (NLM), which designates the processing unit **100** that originates the request, (2) native local hub (NLH), which designates a processing unit **100** that is in the same processing node **202** as the native local master and that may be responsible for transmitting the request to another processing node **202** (a native local master can also be a native local hub), (3) native remote hub (NRH), which designates a processing unit **100** that is in the same plane but a different processing node **202** than the native local master and that is responsible to distribute the request to other processing units **100** in its processing node **202**, (4) native remote leaf (NRL), which designates a processing unit **100** that is in the same plane but a different processing node **202** from the native local master and that is not a native remote hub, (5) foreign local master (FLM), which designates the processing unit **100** that initiates the request within a different plane from the native local master, (6) foreign local hub (NLH), which designates a processing unit **100** that is in the same processing node **202** as the foreign local master and that may be responsible for transmitting the request to another processing node **202** (a foreign local master can also be a foreign local hub), (7) foreign remote hub (FRH), which designates a processing unit **100** that is in the same plane but a different processing node **202** than the foreign master and that is responsible to distribute the request to other processing units **100** in its processing node **202**, and (8) foreign remote leaf (FRL), which designates a processing unit **100** that is in the same plane but a different processing node **202** from the foreign local master and that is not a foreign remote hub.

As shown in FIG. **4**A, the exemplary operation has at least three phases as described above with reference to FIG. **3**, namely, a request (or address) phase, a partial response (Presp) phase, and a combined response (Cresp) phase. These three phases preferably occur in the foregoing order and, from the perspective of an individual processing unit **100**, do not overlap. The operation may additionally have a data phase, which may optionally overlap with any of the request, partial response and combined response phases.

Still referring to FIG. **4**A and referring additionally to FIG. **5**A, the request phase begins when a native local master **100***aap***1***c* (i.e., processing unit **100***c* of processing node **202***aap***1**) performs a synchronized broadcast of a request, for example, a read request, to each of the native local hubs **100***aap***1***a*, **100***aap***1***b*, **100***aap***1***c* and **100***aap***1***d* within its processing node **202***aap***1**. It should be noted that the list of native local hubs includes native local hub **100***aap***1***c*, which is also the native local master. As described further below, this internal transmission is advantageously employed to synchronize the operation of native local hub **100***aap***1***c* with local hubs **100***aap***1***a*, **100***aap***1***b* and **100***aap***1***d* so that the timing constraints discussed below can be more easily satisfied.

In response to receiving the request, each native local hub **100** that is coupled to a native remote hub **100** or foreign local master **100** by its A or B links transmits the operation to its native remote hub(s) and/or foreign local master **100**. Thus, local hubs **100***aap***1***c* and **100***aap***1***d* make no further transmission of the operation on their outbound second tier links. Native local hub **100***aap***1***b* transmits the request via its outbound A link to native remote hub **100***bap***1***a* in processing node **202***bap***1**. Native local hub **100***aap***1***a* transmits the request via its outbound A link to foreign local master **100***aap***2***a* in processing node **202***aap***2**. Each native remote hub **100** receiving the operation in turn transmits the operation to each native remote leaf **100** in its processing node **202**. Thus, for example, native remote hub **100***bap***1***a* transmits the operation to native remote leaves **100***bap***1***b*, **100***bap***1***c* and **100***bap***1***d*.

Distribution of the request within the foreign plane is performed in a similar manner. For example, foreign local master **100***aap***2***a* performs a synchronized broadcast of the request to each of the foreign local hubs **100***aap***2***a*, **100***aap***2***b*, **100***aap***2***c* and **100***aap***2***d* within its processing node **202***aap***2**. In response to receipt of the request, each foreign local hub **100** connected to a foreign remote hub transmits the request to each foreign remote hub **100**. For example, foreign local hub **100***aap***2***b* transmits the request via its outbound A link to foreign remote hub **100***bap***2***a* in processing node **202***bap***2**. Each foreign remote hub **100** receiving the request in turn transmits the request to each foreign remote leaf **100** in its processing node **202**. Thus, for example, foreign remote hub **100***bap***2***a* transmits the operation to foreign remote leaves **100***bap***2***b*, **100***bap***2***c* and **100***bap***2***d*. In this manner, the operation is efficiently broadcast to all processing units **100** within data processing system **500**.

Following the request phase, the partial response (Presp) phase occurs, as shown in FIGS. **4**A and **5**B-**5**C. The partial response phase can be understood as containing at least two subphases, respectively illustrated in FIGS. **5**B-**5**C. In the first subphase of the partial response phase (FIG. **5**B), each remote leaf **100** evaluates the operation and provides its partial response to the operation to its respective remote hub **100**. For example, native remote leaves **100***bap***1***b*, **100***bap***1***c* and **100***bap***1***d* transmit their respective partial responses to native remote hub **100***bap***1***a*, and foreign remote leaves **100***bap***2***b*, **100***bap***2***c* and **100***bap***2***d* transmit their respective partial responses to foreign remote hub **100***bap***2***a*. Each remote hub **100** in turn transmits these partial responses, as well as its own partial response, to a respective local hub **100**. Thus, native remote hub **100***bap***1***a* transmits its collected partial responses to native local hub **100***aap***1***b*, and foreign remote hub **100***bap***2***a* transmits its collected partial responses to foreign remote hub **100***bap***2***a*. Each local hub except for local hubs **100** connected to a processing unit **100** in the other plane then broadcasts its partial response and any other collected partial response(s) to each other local hub **100** within its processing node **202**. For example, all native local hubs **100** except for native local hub **100***aap***1***a* and foreign local hub **100***aap***2***a* broadcasts collected partial responses, if any, as well as their own partial responses, to each other local hub **100** in its processing node **202**. It should be noted by reference to FIG. **5**B that the broadcast of partial responses by the local hubs **100** includes, for timing reasons, the self-broadcast by each local hub **100** of its own partial response. The first subphase of the partial response phase closes with FLM/FLH **100***aap***2***a* transmitting the collective partial response of the second plane to native local hub **100***aap***1***a*.

Referring now to FIG. **5**C, the second phase of the partial response phase begins with native local hub **100***aap***1***a*, which is the native local hub connected to the foreign plane, broadcasting all partial responses it has received, as well as its own partial response, to each of native local hubs **100***aap***1***a*-**100***aap***1***d*. In response to receipt of its self-broadcast of the partial response, native local hub **100***aap***1***a* transmits the partial responses it has received, as well as its own partial response, to foreign local master **100***aap***2***a*. In response thereto, foreign local master **100***aap***2***a* transmits all partial responses it has received, as well as its own partial response, to each of foreign local hubs **100***aap***2***a*-**100***aap***2***d*. At this point, the partial response phase ends with all native and foreign local hubs **100** having received all partial responses within data processing system **500**.

As will be appreciated, the collection of partial responses in the manner shown can be implemented in a number of different ways. For example, it is possible to communicate an individual partial response for each processing unit **100**. Alternatively, for greater efficiency, it may be desirable to accumulate partial responses as they are communicated. In order to ensure that the effect of each partial response is accurately communicated when accumulated in this manner, it is preferred that the partial responses be accumulated, if at all, in a non-destructive manner, for example, utilizing a logical OR function and an encoding in which no relevant information is lost when subjected to such a function (e.g., a "one-hot" encoding).

As further shown in FIG. **4**A and FIG. **5**D, response logic **122** at each local hub **100** compiles all the partial responses its receives to obtain a combined response representing the system-wide response to the request. Local hubs **100***aap***1***a*-**100***aap***1***d* and **100***aap***2***a*-**100***aap***2***d* then self-broadcast the combined response and transmit the combined response to all processing units **100** following the same paths of distribution as employed for the request phase. Thus, the combined response is first broadcast to remote hubs **100**, which in turn transmit the combined response to each remote leaf **100** within their respective processing nodes **202**. For example, native local hub **100***aap***1** transmits the combined response to native remote hub **100***bap***1***a*, which in turn transmits the combined response to native remote leaves **100***bap***1***b*-**100***bap***1***d*. Similarly, foreign local hub **100***aap***1** transmits the combined response to native remote hub **100***bap***2***a*, which in turn transmits the combined response to native remote leaves **100***bap***2***b*-**100***bap***2***d*.

As noted above, servicing the operation may require an additional data phase, such as shown in FIGS. **6**A or **6**B. For example, as shown in FIG. **6**A, if the operation is a read-type operation, such as a read or RWITM operation, foreign remote leaf **100***bap***2***d* may source the requested memory block to local master **100***aap***1***c* via the links connecting foreign remote leaf **100***bap***2***d* to foreign remote hub **100***bap***2***a*, foreign remote hub **100***bap***2***a* to foreign local hub **100***aap***2***b*, and foreign local hub **100***aap***2***b* to native local hub **100***aap***1***a*, and native local hub **100***aap***1***a* to native local master **100***aap***1***c*. Conversely, if the operation is a write-type operation, for example, a cache castout operation writing a modified memory block back to the system memory **132** of foreign remote leaf **100***bap***2***d*, the memory block is transmitted via the links connecting native local master **100***aap***1***c* to native local hub **100***aap***1***a*, native local hub **100***aap***1***a* to foreign local hub **100***aap***2***b*, foreign local hub **100***aap***2***b* to foreign remote hub **100***bap***2***a*, and foreign remote hub **100***bap***2***a* to foreign remote leaf **100***bap***2***d*, as shown in FIG. **6**B.

Referring now to FIG. **4**B, there is illustrated a time-space diagram of an exemplary operation flow of an operation of node-only scope in data processing system 500. In these figures, each processing unit **100** is tagged with a functional identifier indicating its function relative to the other processing units **100** participating in the node-only operation. These functional identifiers include: (1) node master (NM), which designates the processing unit **100** that originates an operation of node-only scope, and (2) node leaf (NL), which designates a processing unit **100** that is in the same processing node **202** as the node master and that is not the node master.

As shown in FIG. **4**B, the exemplary node-only operation has at least three phases as described above: a request (or address) phase, a partial response (Presp) phase, and a combined response (Cresp) phase. Again, these three phases preferably occur in the foregoing order and do not overlap. The operation may additionally have a data phase, which may optionally overlap with any of the request, partial response and combined response phases.

Still referring to FIG. **4**B, the request phase begins when a node master **100***bap***1***a* (i.e., processing unit **100***a* of processing node **202***bap***1**), which functions much like a remote hub in the operational scenario of FIG. **4**A, performs a synchronized broadcast of a request, for example, a read request, to each of the node leaves **100***bap***1***b*, **100***bap***1***c*, and **100***bap***1***d* within its processing node **202***bap***1**. It should be noted that, because the scope of the broadcast transmission is limited to a single node, no internal transmission of the request within node master **100***bap***1***a* is employed to synchronize off-node transmission of the request.

Following the request phase, the partial response (Presp) phase occurs, as shown in FIG. **4**B. In the partial response phase, each of node leaves **100***bap***1***b*, **100***bap***1***c* and **100***bap***1***d* evaluates the operation and provides its partial response to the operation to node master **100***bap***1***a*. Next, as further shown in FIG. **4**B, response logic **122** at node master **100***bap***1***a* within processing node **202***bap***1** compiles the partial responses of the other processing units **100** to obtain a combined response representing the node-wide response to the request. Node master **100***bap***1***a* then broadcasts the combined response to all node leaves **100***bap***1***b*, **100***bap***1***c* and **100***bap***1***d* utilizing the X, Y and Z links of node master **100***bap***1***a*.

As noted above, servicing the operation may require an additional data phase. For example, if the operation is a read-type operation, such as a read or RWITM operation, node leaf **100***bap***1***d* may source the requested memory block to node master **100***bap***1***a* via the Z link connecting node leaf **100***bap***1***d* to node master **100***bap***1***a*. Conversely, if the operation is a write-type operation, for example, a cache castout operation writing a modified memory block back to the system memory **132** of remote leaf **100***bap***1***b*, the memory block is transmitted via the X link connecting node master **100***bap***1***a* to node leaf **100***bap***1***b*.

Of course, the two operations depicted in FIG. **4**A and FIG. **4**B are merely exemplary of the myriad of possible system-wide and node-only operations that may occur concurrently in a multiprocessor data processing system such as data processing system **200**.

V. Exemplary Link Information Allocation

The first tier and second tier links connecting processing units **100** may be implemented in a variety of ways to obtain the topology depicted in FIGS. **2**A-**2**B and to meet the timing constraints illustrated in FIG. **3**. In one preferred embodiment, each inbound and outbound first tier (X, Y and Z) link and each inbound and outbound second tier (A and B) link is implemented as a uni-directional 8-byte bus containing a number of different virtual channels or tenures to convey address, data, control and coherency information.

With reference now to FIGS. **7**A-**7**C, there is illustrated a first exemplary time-sliced information allocation for the first tier X, Y and Z links, intra-plane second tier A and B links, and inter-plane second tier A and B links. As shown, in this first embodiment information is allocated on the first and second tier links in a repeating 8 cycle frame in which the first 4 cycles comprise two address tenures transporting address, coherency and control information and the second 4 cycles are dedicated to a data tenure providing data transport.

Reference is first made to FIG. **7**A, which illustrates the link information allocation for the first tier X, Y and Z links. In each cycle in which the cycle number modulo **8** is 0, byte **0** communicates a transaction type **700***a* (e.g., a read) of a first operation, bytes **1**-**5** provide the 5 lower address bytes **702***a***1** of the request address of the first operation, and bytes **6**-**7** form a reserved field **704**. In the next cycle (i.e., the cycle for which cycle number modulo 8 is 1), bytes **0**-**1** communicate a master tag **706***a* identifying the master **300** of the first operation (e.g., one of L2 cache masters **112** or a master within I/O controller **128**), and byte **2** conveys the high address byte **702***a***2** of the request address of the first operation. A plane indication (e.g., "0" for plane **1** and "1" for plane **2**) identifying the plane originating the first operation is preferably included in one of master tag **706***a* and transaction type **700***a*. Communicated together with this information pertaining to the first operation are up to three additional fields pertaining to different operations, namely, a local partial response **708***a* intended for a local master in the same processing node **202** (bytes **3**-**4**), a combined response **710***a* in byte **5**, and a remote partial response **712***a* intended for a local master in a different processing node **202** (or in the case of a node-only broadcast, the partial response communicated from the node leaf **100** to node master **100**) (bytes **6**-**7**). As noted above, these first two cycles form what is referred to herein as an address tenure.

As further illustrated in FIG. **7**A, the next two cycles (i.e., the cycles for which the cycle number modulo **8** is 2 and 3) form a second address tenure having the same basic pattern as the first address tenure, with the exception that reserved field **704** is replaced with a data tag **714** and data token **715** forming a portion of the data tenure. Specifically, data tag **714** identifies the destination data sink to which the 32 bytes of data payload **716***a*-**716***d* appearing in cycles **4-7** are directed. Its location within the address tenure immediately preceding the payload data advantageously permits the configuration of downstream steering in advance of receipt of the payload data, and hence, efficient data routing toward the specified data sink. Data token **715** provides an indication that a downstream queue entry has been freed and, consequently, that additional data may be transmitted on the paired X, Y, Z or A link without risk of overrun. Again it should be noted that transaction type **700***b*, master tag **706***b*, low address bytes **702***b***1**, and high address byte **702***b***2** all pertain to a second operation, and data tag **714**, local partial response **708***b*, combined response **710***b* and remote partial response **712***b* all relate to one or more operations other than the second operation.

Each transaction type field **700** and combined response field **710** preferably includes a scope indicator **730** indicating whether the operation to which it belongs has a node-only (local) or system-wide (global) scope. As described in greater detail in cross-referenced U.S. patent application Ser. No. 11/055,305, which is incorporated by reference above, data tag **714** further includes a domain indicator **732** that may be set by the LPC to indicate whether or not a remote copy of the data contained within data payload **716***a*-**716***d* may exist.

FIG. **7**B depicts the link information allocation for the intra-plane second tier A and B links. As can be seen by comparison with FIG. **7**A, the link information allocation on the second tier A and B links is the same as that for the first tier links given in FIG. **7**A, except that local partial response fields **708***a*, **708***b* are replaced with reserved fields **718***a*, **718***b*. This replacement is made for the simple reason that, as a second tier link, no local partial responses need to be communicated.

FIG. **7**C depicts the link information allocation for the inter-plane second tier A and B links. As can be seen by comparison with FIG. **7**B, the link information allocation on the inter-plane second tier A and B links is the similar to that for the intra-plane second tier links given in FIG. **7**B, except that (1) reserved fields **718***a*, **718***b* are replaced with foreign-to-native (F-to-N) partial response fields **740***a*, **740***b* for conveying the collective partial response of the foreign plane to the native plane and (2) remote partial response fields **712***a*, **712***b* are replaced with native-to-foreign (N-to-F) partial response fields **742***a*, **742***b* for conveying the collective partial response of the native plan to the foreign plane.

FIG. **8** illustrates an exemplary embodiment of a write request partial response **800**, which may be transported within either a local partial response field **708***a*, **708***b*, a remote partial response field **712***a*, **712***b*, a F-to-N partial response field **740***a*, **740***b*, or N-to-F partial response field **742***a*, **742***b* in response to a write request. As shown, write request partial response **800** is two bytes in length and includes a 15-bit destination tag field **804** for specifying the tag of a snooper (e.g., an IMC snooper **126**) that is the destination for write data and a 1-bit valid (V) flag **802** for indicating the validity of destination tag field **724**.

It will be appreciated by those skilled in the art that the embodiment of FIGS. **7A-7C** depicts only one of a vast number of possible link information allocations. The selected link information allocation that is implemented can be made programmable, for example, through a hardware and/or software-settable mode bit in a configuration register **123** of FIG. **1**. The selection of the link information allocation is typically based on one or more factors, such as the type of anticipated workload. Although the determination of the type(s) of anticipated workload and the setting of configuration register **123** can be performed by a human operator, it is advantageous if the determination is made by hardware and/or software in an automated fashion. For example, in one embodiment, the determination of the type of workload can be made by service processor code executing on one or more of processing units **100** or on a dedicated auxiliary service processor (not illustrated).

VI. Request Phase Structure and Operation

Referring now to FIG. **9**, there is depicted a block diagram illustrating request logic **121***a* within interconnect logic **120** of FIG. **1** utilized in request phase processing of an operation. As shown, request logic **121***a* includes a master multiplexer **900** coupled to receive requests by the masters **300** of a processing unit **100** (e.g., masters **112** within L2 cache **110** and masters within I/O controller **128**). The output of master multiplexer **900** forms one input of a request multiplexer **904**. The second input of request multiplexer **904** is coupled to the output of a remote hub multiplexer **903** having its inputs coupled to the outputs of hold buffers **902***a*, **902***b*, which are in turn coupled to receive and buffer requests on the inbound A and B links, respectively. Remote hub multiplexer **903** implements a fair allocation policy, described further below, that fairly selects among the requests received from the inbound A and B links that are buffered in hold buffers **902***a*-**902***b*. If present, a request presented to request multiplexer **904** by remote hub multiplexer **903** is always given priority by request multiplexer **904**. The output of request multiplexer **904** drives a request bus **905** that is coupled to each of the outbound X, Y and Z links, a node master/remote hub (NM/RH) hold buffer **906**, and the local hub (LH) address launch buffer **910**. A previous request FIFO buffer **907**, which is also coupled to request bus **905**, preferably holds a small amount of address-related information for each of a number of previous address tenures to permit a determination of the address slice or resource bank **1912** to which the address, if any, communicated in that address tenure hashes. For example, in one embodiment, each entry of previous request FIFO buffer **907** contains a “1-hot” encoding identifying a particular one of banks **1912***a*-**1912***n* to which the request address of an associated request hashed. For address tenures in which no request is transmitted on request bus **905**, the 1-hot encoding would be all ‘0’s.

The inbound first tier (X, Y and Z) links are each coupled to the LH address launch buffer **910**, as well as a respective one of node leaf/remote leaf (NL/RL) hold buffers **914***a*-**914***c*. The outputs of NM/RH hold buffer **906**, LH address launch buffer **910**, and NL/RL hold buffers **914***a*-**914***c* all form inputs of a snoop multiplexer **920**. Coupled to the output of LH address launch buffer **910** is another previous buffer **911**, which is preferably constructed like previous request FIFO buffer **907**. The output of snoop multiplexer **920** drives a snoop bus **922** to which tag FIFO queues **924**, the snoopers **304** (e.g., snoopers **116** of L2 cache **110** and snoopers **126** of IMC **124**) of the processing unit **100**, and the outbound A and B links are coupled. Snoopers **304** are further coupled to and supported by local hub (LH) partial response FIFO queues **930** and node master/remote hub (NM/RH) partial response FIFO queue **940**.

Although other embodiments are possible, it is preferable if buffers **902**,**906**, and **914***a*-**914***c* remain short in order to minimize communication latency. In one preferred embodiment, each of buffers **902**, **906**, and **914***a*-**914***c* is sized to hold only the address tenure(s) of a single frame of the selected link information allocation.

With reference now to FIG. **10**, there is illustrated a more detailed block diagram of local hub (LH) address launch buffer **910** of FIG. **9**. As depicted, the local and inbound X, Y and Z link inputs of the LH address launch buffer **910** form inputs of a map logic **1010**, which places each request identified by the plane indication (e.g., within the transaction type **700** or master tag **706** of the request) as originating in the local plane into a position-dependent FIFO queue **1020***a*-**1020***d* corresponding to the particular input on which the request was received. In the depicted nomenclature, the processing unit **100***a* in the upper left-hand corner of a processing node/MCM **202** is the "S" chip; the processing unit **100***b* in the upper right-hand corner of the processing node/MCM **202** is the "T" chip; the processing unit **100***c* in the lower left-hand corner of a processing node/MCM **202** is the "U" chip; and the processing unit **100***d* in the lower right-hand corner of the processing node **202** is the "V" chip. Thus, for example, for native local master/local hub **100***aap***1***c*, in-plane requests received on the local input are placed by map logic **1010** in U FIFO queue **1020***c*, and in-plane requests received on the inbound Y link are placed by map logic **1010** in S FIFO queue **1020***a*. LH address launch buffer **910** further includes a foreign request FIFO queue **1020***e* into which map logic **1010** places all requests received from the other plane (based upon the plane indication provided within transaction type **700** or master tag **706**) while serving as a foreign local hub **100**.

Although placed within position-dependent FIFO queues **1020***a*-**1020***e* requests are not immediately marked as valid and available for dispatch. Instead, the validation of requests in each of position-dependent FIFO queues **1020***a*-**1020***e* is subject to a respective one of programmable delays **1000***a*-**1000***d* in order to synchronize the requests that are received during each address tenure on the four inputs. Thus, the programmable delay **1000***a* associated with the local input, which receives the request self-broadcast at the local master/local hub **100**, is generally considerably longer than those associated with the other inputs. In order to ensure that the appropriate requests are validated, the validation signals generated by programmable delays **1000***a*-**1000***e* are subject to the same mapping by map logic **1010** as the underlying requests.

The outputs of position-dependent FIFO queues **1020***a*-**1020***e* form the inputs of local hub request multiplexer **1030**, which selects one request from among position-dependent FIFO queues **1020***a*-**1020***e* for presentation to local hub request multiplexer **1030** in response to a select signal generated by arbiter **1032**. If an off-plane request is present within foreign request FIFO queue **1020***e*, arbiter **1032** causes local hub request multiplexer **1030** to preferentially output the off-plane request within the next available address tenure of the outbound A link request frame in advance of any in-plane request presented to local hub request multiplexer **1030** by FIFO queues **1020***a*-**1020***d*. Consequently, requests received at a foreign local hub **100** are always non-blocking, and the timing constraints set forth above with respect to FIG. **3** will be satisfied. If no off-plane request is present within foreign request FIFO queue **1020***e*, arbiter **1032** implements a fair arbitration policy that is synchronized in its selections with the arbiters **1032** of all other local hubs **100** within a given processing node **202** so that the same in-plane request is broadcast on the outbound A links at the same time by all local hubs **100** in a processing node **202**, as depicted in FIGS. **4**A and **5**A.

Because the input bandwidth of LH address launch buffer **910** is significantly greater than its output bandwidth, overruns of position-dependent FIFO queues **1020***a*-**1020***d* are a design concern. In a preferred embodiment, queue overruns are prevented by implementing, for each position-dependent FIFO queue **1020***a*-**1020***d*, a pool of local hub tokens equal in size to the depth of the associated position-dependent FIFO queue **1020***a*-**1020***d*. A free local hub token is required for a native local master to send a request to a native local hub and guarantees that the native local hub can queue the request. Thus, a local hub token is allocated when a request is issued by a native local master **100** to a position-dependent FIFO queue **1020***a*-**1020***d* in the native local hub **100** and freed for reuse when arbiter **1032** issues an entry from the position-dependent FIFO queue **1020***a*-**1020***d*. Note that local hub tokens are only used for native plane requests; foreign plane requests overrunning FR FIFO queue **1020***e* are not a concern because the rate of issue of such requests was limited at the native hub launch.

Referring now to FIG. **11**, there is depicted a more detailed block diagram of tag FIFO queues **924** of FIG. **9**. As shown, tag FIFO queues **924** include a local hub (LH) tag FIFO queue **924***a*, remote hub (RH) tag FIFO queues **924***b***0**-**924***b***1**, node master (NM) tag FIFO queue **924***b***2**, remote leaf (RL) tag FIFO queues **924***c***0**-**924***c***1**, **924***d***0**-**924***d***1** and **924***e***0**-**924***e***1**, and node leaf (NL) tag FIFO queues **924***c***2**, **924***d***2** and **924***e***2**. The master tag of a request of an operation of system-wide scope is deposited in each of tag FIFO queues **924***a*, **924***b***0**-**924***b***1**, **924***c***0**-**924***c***1**, **924***d***0**-**924***d***1** and **924***e***0**-**924***e***1** when the request is received at the processing unit(s) **100** serving in each of these given roles (LH, RH, and RL) for that particular request. Similarly, the master tag of a request of an operation of node-only scope is deposited in each of tag FIFO queues **924***b***2**, **924***c***2**, **924***d***2** and **924***e***2** when the request is received at the processing unit(s) **100** serving in each of these given roles (NM and NL) for that particular request. The master tag is retrieved from each of tag FIFO queues **924** when the combined response is received at the associated processing unit **100**. Thus, rather than transporting the master tag with the combined response, master tags are retrieved by a processing unit **100** from its tag FIFO queue **924** as needed, resulting in bandwidth savings on the first and second tier links. Given that the order in which a combined response is received at the various processing units **100** is identical to the order in which the associated request was received, a FIFO policy for allocation and retrieval of the master tag can advantageously be employed.

LH tag FIFO queue **924***a* includes a number of entries, each including a master tag field **1100** for storing the master tag of a request launched by arbiter **1032**. Each of tag FIFO queues **924***b***0**-**924***b***1** similarly includes multiple entries, each including at least a master tag field **1100** for storing the master tag of a request of system-wide scope received by a remote hub **100** via a respective one of the inbound A and B links. Tag FIFO queues **924***c***0**-**924***c***1**, **924***d***0**-**924***d***1** and **924***e***0**-**924***e***1** are similarly constructed and each hold master tags of requests of system-wide scope received by a remote leaf **100** via a unique pairing of inbound first and second tier links. For requests of node-only broadcast scope, NM tag FIFO queues **924***b***2** holds the master tags of requests originated by the node master **100**, and each of NL tag FIFO queues **924***c***2**, **924***d***2** and **924***e***2** provides storage for the master tags of requests received by a node leaf **100** on a respective one of the first tier X, Y and Z links.

As depicted in FIG. **13**, which is described below, entries within LH tag FIFO queue **924***a* have the longest tenures for system-wide broadcast operations, and NM tag FIFO queue **924***b***2** have the longest tenures for node-only broadcast operations. Consequently, the depths of LH tag FIFO queue **924***a* and NM tag FIFO queue **924***b***2** respectively limit the number of concurrent operations of system-wide scope that a processing node **202** can issue on the interconnect fabric and the number of concurrent operations of node-only scope that a given processing unit **100** can issue on the interconnect fabric. These depths have no necessary relationship and may be different. However, the depths of tag FIFO queues **924*b*0**-**924*b*1**, **924*c*0**-**924*c*1**, **924*d*0**-**924*d*1** and **924*e*0**-**924*e*1** are preferably designed to be equal to that of LH tag FIFO queue **924*a***, and the depths of tag FIFO queues **924*c*2**, **924*d*2** and **924*e*2** are preferably designed to be equal to that of NM tag FIFO queue **924*b*2**.

With reference now to FIGS. **12**A and **12**B, there are illustrated more detailed block diagrams of exemplary embodiments of the local hub (LH) partial response FIFO queue **930** and node master/remote hub (NM/RH) partial response FIFO queue **940** of FIG. **9**. As indicated, LH partial response FIFO queue **930** includes a number of entries **1200** that each includes a partial response field **1202** for storing an accumulated partial response for a request and a response flag array **1204** having respective flags for each of the 6 possible sources from which the local hub **100** may receive a partial response (i.e., local (L), first tier X, Y, Z links, and second tier A and B links) at different times or possibly simultaneously. In addition, each entry **1200** includes a foreign/native (F/N) flag **1206** indicating whether the local hub is a foreign local hub FLH **100** or a native local hub (NLH) **100**. Entries **1200** within LH partial response FIFO queue **930** are allocated via an allocation pointer **1210** and deallocated via a deallocation pointer **1212**. Various flags comprising response flag array **1204** are accessed utilizing A pointer **1214**, B pointer **1215**, X pointer **1216**, Y pointer **1218**, and Z pointer **1220**.

As described further below, when a partial response for a particular request is received by partial response logic **121*b*** at a local hub **100**, the partial response is accumulated within partial response field **1202**, and the link from which the partial response was received is recorded by setting the corresponding flag within response flag array **1204**. The corresponding one of pointers **1214**, **1215**, **1216**, **1218** and **1220** is then advanced to the subsequent entry **1200**.

Of course, as described above, each processing unit **100** need not be fully coupled to other processing units **100** by each of its 5 inbound (X, Y, Z, A and B) links. Accordingly, flags within response flag array **1204** that are associated with unconnected links are ignored. The unconnected links, if any, of each processing unit **100** may be indicated, for example, by the configuration indicated in configuration register **123**, which may be set, for example, by boot code at system startup or by the operating system when partitioning data processing system **200**.

As can be seen by comparison of FIG. **12**B and FIG. **12**A, NM/RH partial response FIFO queue **940** is constructed similarly to LH partial response FIFO queue **930**. NM/RH partial response FIFO queue **940** includes a number of entries **1230** that each includes a partial response field **1202** for storing an accumulated partial response and a response flag array **1234** having respective flags for each of the up to 4 possible sources from which the node master or remote hub **100** may receive a partial response (i.e., node master (NM)/remote (R), and first tier X, Y, and Z links). In addition, each entry **1230** includes a route field **1236** identifying whether the operation is a node-only or system-wide broadcast operation and, for system-wide broadcast operations, which of the inbound second tier links the request was received upon (and thus which of the outbound second tier links the accumulated partial response will be transmitted on). Entries **1230** within NM/RH partial response FIFO queue **940** are allocated via an allocation pointer **1210** and deallocated via a deallocation pointer **1212**. Various flags comprising response flag array **1234** are accessed and updated utilizing X pointer **1216**, Y pointer **1218**, and Z pointer **1220**.

As noted above with respect to FIG. **12**A, each processing unit **100** need not be fully coupled to other processing units **100** by each of its first tier X, Y, and Z links. Accordingly, flags within response flag array **1204** that are associated with unconnected links are ignored. The unconnected links, if any, of each processing unit **100** may be indicated, for example, by the configuration indicated in configuration register **123**.

Referring now to FIG. **13**, there is depicted a time-space diagram illustrating the tenure in one plane of an exemplary system-wide broadcast operation with respect to the exemplary data structures depicted in FIG. **9** through FIG. **12**B. As shown at the top of FIG. **13** and as described previously with reference to FIG. **4**A, the operation is issued by a native or foreign local master **100** to each native or foreign local hub **100** (only one of which is shown). The native or foreign local hub **100** forwards the operation to a native or foreign remote hub **100**, which in turn forwards the operation to its remote leaves **100** (only one of which is shown). The partial responses to the operation traverse the same series of links in reverse order back to the native or foreign local hubs **100**, which perform intra-node and inter-plane exchange of the combined responses, as previously described. The native or foreign local hubs **100** then generate and distribute the combined response following the same transmission paths as the request. Thus, the native or foreign local hubs **100** transmit the combined response to the associated native or foreign remote hubs **100**, which transmits the combined response to the attached native or foreign remote leaves **100**.

As dictated by the timing constraints described above, the time from the initiation of the operation by the native or foreign local master **100** to its launch by the attached local hubs **100** is a variable time, the time from the launch of the operation by local hubs **100** to its receipt by the remote leaves **100** is a bounded time, the partial response latency from the remote leaves **100** to the local hubs **100** is a variable time, and the combined response latency from the local hubs **100** to the remote leaves **100** is a bounded time.

Against the backdrop of this timing sequence, FIG. **13** illustrates the tenures of various items of information within various data structures within data processing system **200** during the request phase, partial response phase, and combined response phase of an operation. In particular, the tenure of a request in a LH launch buffer **910** (and hence the tenure of a local hub token) is depicted at reference numeral **1300**, the tenure of an entry in LH tag FIFO queue **924*a*** is depicted at reference numeral **1302**, the tenure of an entry **1200** in LH partial response FIFO queue **930** is depicted at block **1304**, the tenure of an entry in a RH tag FIFO **924*b*0** or **924*b*1** is depicted at reference numeral **1306**, the tenure of an entry **1230** in a NM/RH partial response FIFO queue **940** is depicted at reference numeral **1308**, and the tenure of entries in RL tag FIFO queues **924*c*0**-**924*c*1**, **924*d*0**-**924*d*1** and **924*e*0**-**924*e*1** is depicted at reference numeral **1310**. FIG. **13** further illustrates the duration of a protection window **1312*a*** and window extension **1312*b*** (also **312*a*-312*b*** of FIGS. **3** and **6**) extended by the snooper within remote leaf **100** to protect the transfer of coherency ownership of the memory block to local master **100** from generation of its partial response until after receipt of the combined response. As shown at reference numeral **1314** (and also at **313** of FIG. **3**), local master **100** also protects the transfer of ownership from receipt of the combined response.

As indicated at reference numerals **1302**, **1306** and **1310**, the entries in the LH tag FIFO queue **924*a***, RH tag FIFO queues **924*b*0**-**924*b*1** and RL tag FIFO queues **924*c*0**-**924*c*1**, **924*d*0**-**924*d*1** and **924*e*0**-**924*e*1** are subject to the longest tenures. Consequently, the minimum depth of tag FIFO queues **924** (which are generally designed to be the same) limits the maximum number of requests that can be in flight in data processing system **200** at any one time. In general, the desired depth of tag FIFO queues **924** can be selected by dividing the expected maximum latency from snooping of a request by an arbitrarily selected processing unit **100** to receipt of the combined response by that processing unit **100** by the maximum number of requests that can be issued given the selected link information allocation. Although the other queues (e.g., LH partial response FIFO queue **930** and NM/RH partial response FIFO queue **940**) may safely be assigned shorter queue depths given the shorter tenure of their entries, for simplicity it is desirable in at least some embodiments to set the depth of LH partial response FIFO queue **930** to be the same as tag FIFO queues **924**, and to set the depth of NM/RH partial response FIFO queue **940** to be equal to the depth of NM tag FIFO **924*b*2** plus t2/2 times the depth of RL tag FIFO queues **924**.

With reference now to FIG. **14**A-**14**F, flowcharts are given that respectively depict exemplary processing of an operation during the request phase at a native local master (or node master), native local hub, native or foreign remote hub (or node master), native or foreign remote leaf (or node leaf), foreign local master, and foreign local hub in accordance with an exemplary embodiment of the present invention. Referring now specifically to FIG. **14**A, request phase processing at the native local master (or node master, if a node-only broadcast) **100** begins at block **1400** with the generation of a request by a particular master **300** (e.g., one of masters **112** within an L2 cache **110** or a master within an I/O controller 128) within a native local master (or node master) **100**. Following block **1400**, the process proceeds to blocks **1402**, **1404**, **1406**, and **1408**, each of which represents a condition on the issuance of the request by the particular master **300**. The conditions illustrated at blocks **1402** and **1404** represent the operation of master multiplexer **900**, and the conditions illustrated at block **1406** and **1408** represent the operation of request multiplexer **904**.

Turning first to blocks **1402** and **1404**, master multiplexer **900** outputs the request of the particular master **300** if the fair arbitration policy governing master multiplexer **900** selects the request of the particular master **300** from the requests of (possibly) multiple competing masters **300** (block **1402**) and, if the request is a system-wide broadcast, if a local hub token is available for assignment to the request (block **1404**). As indicated by block **1415**, if the master **300** selects the scope of its request to have a node-only scope (for example, by reference to a setting of configuration register **123** and/or a scope prediction mechanism, such as that described in above-referenced U.S. patent application Ser. No. 11/055,305, no local hub token is required, and the condition illustrated at block **1404** is omitted.

Assuming that the request of the particular master **300** progresses through master multiplexer **900** to request multiplexer **904**, request multiplexer **904** issues the request on request bus **905** only if a address tenure is then available for a request in the outbound first tier link information allocation (block **1406**). That is, the output of request multiplexer **904** is timeslice aligned with the selected link information allocation and will only generate an output during cycles designed to carry a request (e.g., cycle **0** or **2** of the embodiment of FIG. 7A). As further illustrated at block **1408**, request multiplexer **904** will only issue a request if no request from the inbound second tier A and B links is presented by remote hub multiplexer **903** (block **1406**), which is always given priority. Thus, the second tier links are guaranteed to be non-blocking with respect to inbound requests. Even with such a non-blocking policy, requests by masters **300** can prevented from "starving" through implementation of an appropriate policy in the arbiter **1032** of the upstream hubs that prevents "brickwalling" of requests during numerous consecutive address tenures on the inbound A and B link of the downstream hub.

If a negative determination is made at any of blocks **1402**-**1408**, the request is delayed, as indicated at block **1410**, until a subsequent cycle during which all of the determinations illustrated at blocks **1402**-**1408** are positive. If, on the other hand, positive determinations are made at all of blocks **1402**-**1408**, the process proceeds to block **1417**. Block **1417** represents that requests of node-only scope (as indicated by scope indicator **730** of Ttype field **700**) are subject to two additional conditions illustrated at blocks **1419** and **1423**. First, as shown at block **1419**, if the request is a node-only broadcast request, request multiplexer **904** will issue the request only if an entry is available for allocation to the request in NM tag FIFO queue **924*b*2**. If not, the process passes from block **1419** to block **1410**, which has been described.

Second, as depicted at block **1423**, request multiplexer **904** will issue a request of node-only scope only if the request address does not hash to the same bank **1912** of a banked resource **1910** as any of a selected number of prior requests buffered within previous request FIFO buffer **907**. For example, assuming that a snooping device **1900** and its associated resource **1910** are constructed so that snooping device **1900** cannot service requests at the maximum request arrival rate, but can instead service requests at a fraction of the maximum arrival rate expressed as 1/R, the selected number of prior requests with which the current node-only request vying for launch by request multiplexer **904** is compared to determine if it falls in the same address slice is preferably R-1. If multiple different snooping devices **1900** are to be protected in this manner from request overrun, the selected number of requests R-1 is preferably set to the maximum of the set of quantities R-1 calculated for the individual snooping devices **1900**. Because processing units **100** preferably do not coordinate their selection of requests for broadcast, the throttling of requests in the manner illustrated at block **1423** does not guarantee that the arrival rate of requests at a particular snooping device **1900** will not exceed the service rate of the snooping device **1900**. However, the throttling of node-only broadcast requests in the manner shown will limit the number of requests that can arrive in a given number of cycles, which can be expressed as:

throttled_arr_rate=PU requests per R cycles

where PU is the number of processing units **100** per processing node **202**. Snooping devices **1900** are preferably designed to handle node-only broadcast requests arriving at such a throttled arrival rate without retry.

If the condition shown at block **1423** is not satisfied, the process passes from block **1423** to block **1410**, which has been described. However, if both of the conditions illustrated at blocks **1419** and **1423** are satisfied, request multiplexer **904** issues the node-only broadcast request on request bus **905**, and the process passes through page connector **1425** to block **1427** of FIG. **14**C.

Returning again to block **1417**, if the request is system-wide broadcast request rather than a node-only broadcast request, the process proceeds to block **1412**, beginning tenure **1300** of FIG. **13**. Block **1412** depicts request multiplexer **904** broadcasting the request on request bus **905** to each of the outbound X, Y and Z links and to the local hub address launch buffer **910**. Thereafter, the process bifurcates and passes through page connectors **1414** and **1416** to FIG. **14**B, which illustrates the processing of the request at each of the native local hubs **100**.

With reference now to FIG. **14**B, processing of the request at the native local hub **100** that is also the native local master **100** is illustrated beginning at block **1416**, and processing of the request at each of the other native local hubs **100** in the same processing node **202** as the native local master **100** is depicted beginning at block **1414**. Turning first to block **1414**, requests received by a native local hub **100** on the inbound X, Y and Z links are received by LH address launch buffer **910**. As depicted at block **1420** and in FIG. **10**, map logic **1010** maps each of the X, Y and Z requests to the appropriate ones of position-dependent FIFO queues **1020***a*-**1020***d* for buffering. As noted above, requests received on the X, Y and Z links and placed within position-dependent queues **1020***a*-**1020***d* are not immediately validated. Instead, the requests are subject to respective ones of tuning delays **1000***a*-**1000***d*, which synchronize the handling of the X, Y and Z requests and the local request on a given native local hub **100** with the handling of the corresponding requests at the other native local hubs **100** in the same processing node **202** (block **1422**). Thereafter, as shown at block **1430**, the tuning delays **1000** validate their respective requests within position-dependent FIFO queues **1020***a*-**1020***d*.

Referring now to block **1416**, at the native local master/native local hub **100**, the request on request bus **905** is fed directly into LH address launch buffer **910**. Because no inter-chip link is traversed, this local request arrives at LH address launch FIFO **910** earlier than requests issued in the same cycle arrive on the inbound X, Y and Z links. Accordingly, following the mapping by map logic **1010**, which is illustrated at block **1424**, one of tuning delays **1000***a*-**100***d* applies a long delay to the local request to synchronize its validation with the validation of requests received on the inbound X, Y and Z links (block **1426**). Following this delay interval, the relevant tuning delay **1000** validates the local request, as shown at block **1430**.

Following the validation of the requests queued within LH address launch buffer **910** at block **1430**, the process then proceeds to blocks **1434**-**1440**, each of which represents a condition on the issuance of a request from LH address launch buffer **910** enforced by arbiter **1032**. As noted above, the arbiters **1032** within all processing units **100** are synchronized so that the same decision is made by all native local hubs **100** without inter-communication. As depicted at block **1434**, an arbiter **1032** permits local hub request multiplexer **1030** to output a request only if an address tenure is then available for the request in the outbound second tier link information allocation. Thus, for example, arbiter **1032** causes local hub request multiplexer **1030** to initiate transmission of requests only during cycle **0** or **2** of the embodiment of FIG. **7**B. In addition, an in-plane request is output by local hub request multiplexer **1030** if the fair arbitration policy implemented by arbiter **1032** determines that no foreign request is pending in foreign request FIFO **1020***e* and the in-plane request belongs to the position-dependent FIFO queue **1020***a*-**1020***d* that should be serviced next according to the fair allocation policy of arbiter **1032** (block **1436**).

As depicted further at blocks **1437** and **1438**, arbiter **1032** causes local hub request multiplexer **1030** to output a request only if it determines that it has not been outputting too many requests in successive address tenures. Specifically, at shown at block **1437**, to avoid overdriving the request buses **905** of the hubs **100** connected to the outbound A and B links, arbiter **1032** assumes the worst case (i.e., that traffic from the other plane will consume half of the available bandwidth and that the upstream hub **100** connected to the other second tier link of the downstream hub **100** will consume half of the remaining bandwidth) and accordingly launches requests during no more than one-fourth (i.e., 1/(no. of planes*t**2**)) of the available address tenures. In addition, as depicted at block **1438**, arbiter **1032** further restricts the launch of requests below a fair allocation of the traffic on the second tier links to avoid possibly "starving" the masters **300** in the processing units **100** coupled to its outbound A and B links.

For example, given the embodiment of FIGS. **2**A-**2**B, where there are 2 pairs of second tier links and 4 processing units **100** per processing node **202**, traffic on the request bus **905** of the downstream hub **100** is subject to contention by traffic from the other plane plus up to 9 in-plane processing units **100**, namely, the 4 processing units **100** in each of the 2 processing nodes **202** coupled to the downstream hub **100** by second tier links and the downstream hub **100** itself. Consequently, an exemplary fair allocation policy that divides the bandwidth of request bus **905** evenly among the possible request sources allocates 4/18 of the bandwidth to each of the inbound A and B links, 1/18 of the bandwidth to the local masters **300** and the remaining ½ of the bandwidth to requests received from another plane. Generalizing for any number of first and second tier links, the fraction of the available address frames allocated consumed by the exemplary fair allocation policy employed by arbiter **1032** can be expressed as:

$$\text{fraction}=(t1/2+1)/[(t2/2*(t1/2+1)+1)*p]$$

where t**1** and t**2** represent the total number of first and second tier links to which a processing unit **100** may be coupled, the quantity "t**1**/2+1" represents the number of processing units **100** per processing node **202**, the quantity "t**2**/2" represents the number of processing nodes **202** to which a downstream hub **100** may be coupled, the constant quantity "1" represents the fractional bandwidth allocated to the downstream hub **100**, and p is the number of planes.

As shown at block **1439**, arbiter **1032** further throttles the transmission of system-wide broadcast requests by issuing a system-wide broadcast request only if the request address does not hash to the same bank **1912** of a banked resource **1910** as any of a R-1 prior requests buffered within previous request FIFO buffer **911**, where 1/R is the fraction of the maximum arrival rate at which the slowest protected snooping device **1900** can service requests. Thus, the throttling of system-wide broadcast requests in the manner shown will limit the number of requests that can arrive at a given snooping device **1900** in a given number of cycles, which can be expressed as:

$$\text{throttled\_arr\_rate}=N \text{ requests per } R \text{ cycles}$$

where N is the number of processing nodes **202**. Snooping devices **1900** are preferably designed to handle requests arriving at such a throttled arrival rate without retry.

Referring finally to the condition shown at block **1440**, arbiter **1032** permits an in-plane request to be output by local hub request multiplexer **1030** only if an entry is available for allocation in LH tag FIFO queue **924***a* (block **1440**). In order to preserve non-blocking flow for requests received from another plane, preferably only one-half of the entries within LH tag FIFO queue **924***a* are made available for in-plane requests.

If a negative determination is made at any of blocks **1434**-**1440**, the request is delayed, as indicated at block **1442**, until a subsequent cycle during which all of the determinations illustrated at blocks **1434**-**1440** are positive. If, on the other hand, positive determinations are made at all of blocks **1434**-**1440**, arbiter **1032** signals local hub request multiplexer **1030** to output the selected in-plane request to an input of multiplexer **920**, which always gives priority to a request, if any, presented by LH address launch buffer **910**. Thus, multiplexer **920** issues the request on snoop bus **922**. It should be noted that the other ports of multiplexer **920** (e.g., RH, RLX, RLY, and RLZ) could present requests concurrently with LH address launch buffer **910**, meaning that the maximum bandwidth of snoop bus **922** must equal 10/8 (assuming the embodiment of FIG. 7B) of the bandwidth of the outbound A and B links in order to keep up with maximum arrival rate.

It should also be observed that only requests buffered within local hub address launch buffer **910** are transmitted on the outbound A and B links and are required to be aligned with address tenures within the link information allocation. Because all other requests competing for issuance by multiplexer **920** target only the local snoopers **304** and their respective FIFO queues rather than the outbound A and B links, such requests may be issued in the remaining cycles of the information frames. Consequently, regardless of the particular arbitration scheme employed by multiplexer **920**, all requests concurrently presented to multiplexer **920** are guaranteed to be transmitted within the latency of a single information frame.

As indicated at block **1444**, in response to the issuance of the request on snoop bus **922**, LH tag FIFO queue **924***a* records the master tag specified in the request in the master tag field **1100** of the next available entry, beginning tenure **1302**. The request is then routed to the outbound A and B links, as shown at block **1446**. The process then passes through page connector **1448** to FIG. **14**C, which depicts the processing of the request at each of the native remote hubs **100** during the request phase. For native local hubs **100** coupled by a second tier link to a foreign local master **100**, processing also passes through page connector **1449** to FIG. **14**E, which illustrates processing of the request at the foreign local master **100** during the request phase.

The process depicted in FIG. **14**B also proceeds from block **1446** to block **1450**, which illustrates native local hub **100** freeing the local hub token allocated to the request in response to the removal of the request from LH address launch buffer **910**, ending tenure **1300**. The request is further routed to the snoopers **304** in the native local hub **100**, as shown at block **1452**. In response to receipt of the request, snoopers **304** generate a partial response (block **1454**), which is recorded within LH partial response FIFO queue **930**, beginning tenure **1304** (block **1456**). In particular, at block **1456**, an entry **1200** in the LH partial response FIFO queue **930** is allocated to the request by reference to allocation pointer **1210**, allocation pointer **1210** is incremented, the partial response of the local hub is placed within the partial response field **1202** of the allocated entry, and the local (L) flag is set in the response flag field **1204**. In addition, the F/N flag **1206** of the entry **1200** is set to indicate "native", signifying that the processing unit is serving as a native local hub **100** for the operation. Thereafter, request phase processing at the native local hub **100** ends at block **1458**.

Referring now to FIG. **14**C, there is depicted a high level logical flowchart of an exemplary method of request processing at a native or foreign remote hub (or for a node-only broadcast request, a node master) **100** in accordance with the present invention. As depicted, for a system-wide broadcast request, the process begins at page connector **1448** upon receipt of the request at the native or foreign remote hub **100** on one of its inbound A and B links. As noted above, after the request is latched into a respective one of hold buffers **902***a*-**902***b* as shown at block **1460**, the request is evaluated by remote hub multiplexer **903** and request multiplexer **904** for transmission on request bus **905**, as depicted at blocks **1464** and **1465**. Specifically, at block **1464**, remote hub multiplexer **903** determines whether to output in-plane requests in accordance with a fair allocation policy that evenly allocates address tenures to in-plane requests received on the inbound second tier links. Foreign requests originating in a different plane, however, are preferably non-blocking and not subject to a fair allocation policy. In addition, at illustrated at block **1465**, request multiplexer **904**, which is timeslice-aligned with the first tier link information allocation, outputs a request only if an address tenure is then available. Thus, as shown at block **1466**, if a request is not a winning request under the policy of multiplexer **903** or if no address tenure is then available, multiplexer **904** waits for the next address tenure. It will be appreciated, however, that even if a request received on an inbound second tier link is delayed, the delay will be no more than one frame of the first tier link information allocation.

If both the conditions depicted at blocks **1464** and **1465** are met, multiplexer **904** launches the request on request bus **905**, and the process proceeds from block **1465** to block **1468**. As indicated, request phase processing of node-only broadcast operations at the node master **100**, which continues at block **1423** from block **1421** of FIG. **14**A, also passes to block **1468**. Block **1468** illustrates the routing of the request issued on request bus **905** to the outbound X, Y and Z links, as well as to NM/RH hold buffer **906**. Following block **1468**, the process bifurcates. A first path passes through page connector **1470** to FIG. **14**D, which illustrates an exemplary method of request processing at the native or foreign remote (or node) leaves **100**. The second path from block **1468** proceeds to block **1474**, which illustrates the snoop multiplexer **920** determining which of the requests presented at its inputs to output on snoop bus **922**. As indicated, snoop multiplexer **920** prioritizes local hub requests over remote hub requests, which are in turn prioritized over requests buffered in NL/RL hold buffers **914***a*-**914***c*. Thus, if a local hub request is presented for selection by LH address launch buffer **910**, the request buffered within NM/RH hold buffer **906** is delayed, as shown at block **1476**. If, however, no request is presented by LH address launch buffer **910**, snoop multiplexer **920** issues the request from NM/RH hold buffer **906** on snoop bus **922**.

In response to detecting the request on snoop bus **922**, the appropriate one of tag FIFO queues **924***b* (i.e., for node-only broadcast requests, NM tag FIFO queue **924***b***2**, and for system-wide broadcast request, the one of RH tag FIFO queues **924***b***0** and **924***b***1** associated with the inbound second tier link on which the request was received) places the master tag specified by the request into master tag field **1100** of its next available entry, beginning tenure **1306** or **1320** (block **1478**). As noted above, node-only broadcast requests and system-wide broadcast requests are differentiated by a scope indicator **730** within the Ttype field **700** of the request. The request is further routed to the snoopers **304** in the native or foreign remote hub (or node master) **100**, as shown at block **1480**. In response to receipt of the request, snoopers **304** generate a partial response at block **1482**, which is recorded within NM/RH partial response FIFO queue **940**, beginning tenure **1308** or **1322** (block **1484**). In particular, an entry **1230** in the NM/RH partial response FIFO queue **940** is allocated to the request by reference to its allocation pointer **1210**, the allocation pointer **1210** is incremented, the partial response of the remote hub is placed within the partial response field **1202**, and the node master/remote flag (NM/R) is set in the response flag field **1234**. It should be noted that NM/RH partial response FIFO queue **940** thus buffers partial responses for operations of differing scope in the same data structure. Thereafter, request phase processing at the remote hub **100** ends at block **1486**.

With reference now to FIG. **14**D, there is illustrated a high level logical flowchart of an exemplary method of request processing at a native or foreign remote leaf (or node leaf) **100** in accordance with the present invention. As shown, the process begins at page connector **1470** upon receipt of the request at the native or foreign remote leaf or node leaf **100** on one of its inbound X, Y and Z links. As indicated at block **1490**, in response to receipt of the request, the request is latched into of the particular one of NL/RL hold buffers **914***a*-**914***c* associated with the first tier link upon which the request was received. Next, as depicted at block **1491**, the request is evaluated by snoop multiplexer **920** together with the other requests presented to its inputs. As discussed above, snoop multiplexer **920** prioritizes local hub requests over remote hub requests, which are in turn prioritized over requests buffered in NL/RL hold buffers **914***a*-**914***c*. Thus, if a local hub or remote hub request is presented for selection, the request buffered within the NL/RL hold buffer **914** is delayed, as shown at block **1492**. If, however, no higher priority request is presented to snoop multiplexer **920**, snoop multiplexer **920** issues the request from the NURL hold buffer 914 on snoop bus **922**, fairly choosing between X, Y and Z requests.

In response to detecting request on snoop bus **922**, the particular one of tag FIFO queues **924***c***0**-**924***c***2**, **924***d***0**-**924***c***2** and **924***e***0**-**924***e***2** associated with the scope of the request and the by which the request was received places the master tag specified by the request into the master tag field **1100** of its next available entry, beginning tenure **1310** or **1324** (block **1493**). That is, the scope indicator **730** within the Ttype field **700** of the request is utilized to determine whether the request is of node-only or system-wide scope. As noted above, for node-only broadcast requests, the particular one of NL tag FIFO queues **924***c***2**, **924***d***2** and **924***e***2** associated with the inbound first tier link upon which the request was received buffers the master tag. For system-wide broadcast requests, the master tag is placed in the particular one of RL tag FIFO queues **924***c***0**-**924***c***1**, **924***d***0**-**924***d***1** and **924***e***0**-**924***e***1** corresponding to the combination of inbound first and second tier links upon which the request was received. The request is further routed to the snoopers **304** in the native or foreign remote leaf (or node leaf) **100**, as shown at block **1494**. In response to receipt of the request, the snoopers **304** process the request, generate their respective partial responses, and accumulate the partial responses to obtain the partial response of that processing unit **100** (block **1495**). As indicated by page connector **1497**, the partial response of the snoopers **304** of the native or foreign remote leaf or node leaf **100** is handled in accordance with FIG. **16**A, which is described below.

Referring now to FIG. **14**E, there is depicted a high level logical flowchart of an exemplary method of request processing at a foreign local master **100** in accordance with the present invention. As depicted, the process begins at page connector **1449** upon receipt of a system-wide broadcast request at the foreign local master **100** on one of its inbound A and B links. As noted above, after the request is latched into a respective one of hold buffers **902***a*-**902***b* as shown at block **1429**, the request is evaluated by remote hub multiplexer **903** and request multiplexer **904** for transmission on request bus **905**, as depicted at blocks **1431** and **1433**. Specifically, at block **1431**, remote hub multiplexer **903** selects the inbound foreign request as the winning request over any competing in-plane request received on the other inbound second tier link. In addition, at illustrated at block **1433**, request multiplexer **904**, which is timeslice-aligned with the first tier link information allocation, outputs a request only if an address tenure is then available. Thus, as shown at block **1435**, if no address tenure is then available, multiplexer **904** waits for the next address tenure.

If both the conditions depicted at blocks **1431** and **1433** are met, multiplexer **904** launches the request on request bus **905**, and the process proceeds from block **1433** to block **1441**. Block **1441** illustrates the routing of the foreign request issued on request bus **905** to the outbound X, Y and Z links, as well as to foreign request FIFO queue **1020***e* of FIG. **10**. Following block **1441**, the process bifurcates. A first path passes through page connector **1443** to FIG. **14**F, which illustrates an exemplary method of request processing at the foreign local master **100** when serving as a foreign local hub **100**. The second path from block **1441** passes through page connector **1445** to FIG. **14**F, which also illustrates an exemplary method of request processing at a foreign local hub **100** other than the foreign local master **100**.

With reference to now to FIG. **14**F, there is illustrated a high level logical flowchart of an exemplary method of request phase processing at a foreign remote hub **100** in accordance with the present invention. The process begins at either page connectors **1443** or **1445** in response to receipt of a foreign request within LH address launch buffer **910** either via request bus **905** or one of the inbound first tier X, Y and Z links. As described above, map logic **1010** maps the foreign request into foreign request FIFO queue **1020***e*, as depicted at block **1447**. As discussed above, the validation of the foreign request in foreign request FIFO queue **1020***e* at the foreign local master/foreign local hub **100** is subject to a delay **1000** (blocks **1451**-**1453**) equal to a first tier link latency to synchronize the launch of the request onto snoop bus **922** at all foreign local hubs **100**. At foreign local hubs **100** other than the foreign local master **100**, only a small tuning delay is applied to achieve synchronization.

As depicted at blocks **1455**-**1457**, arbiter **1032** permits local hub request multiplexer **1030** to output a foreign request only if an address tenure is then available for the request in the outbound second tier link information allocation. Thus, for example, arbiter **1032** causes local hub request multiplexer **1030** to initiate transmission of requests only during cycle **0** or **2** of the link information allocation embodiment of FIG. **7**B and to wait otherwise. As indicated at block **1459**, in response to the issuance of the request on snoop bus **922**, LH tag FIFO queue **924***a* records the master tag specified in the request in the master tag field **1100** of the next available entry, beginning tenure **1302**. The request is then routed to the outbound A and B links, as shown at block **1461**. The process then passes through page connector **1448** to FIG. **14**B, which depicts the processing of the request at each of the foreign remote hubs **100** during the request phase.

The process depicted in FIG. **14**F also proceeds from block **1461** to block **1465**, which illustrates foreign local hub **100** routing the request to the snoopers **304** in the foreign local hub **100**. In response to receipt of the request, snoopers **304** generate a partial response (block **1467**), which is recorded within LH partial response FIFO queue **930**, beginning tenure **1304** (block **1469**). In particular, at block **1456**, an entry **1200** in the LH partial response FIFO queue **930** is allocated to the request by reference to allocation pointer **1210**, allocation pointer **1210** is incremented, the partial response of the local hub is placed within the partial response field **1202** of the allocated entry, and the local (L) flag is set in the response flag field **1204**. In addition, the F/N flag **1206** of the entry **1200** is set to indicate “foreign”, signifying that the processing unit is serving as a foreign local hub **100** for the operation. Thereafter, request phase processing at the native local hub **100** ends at block **1471**.

FIG. **14**G is a high level logical flowchart of an exemplary method by which snooper s **304** generate partial responses for requests, for example, at blocks **1454**, **1467**, **1482** and **1495** of FIGS. **14**B-**14**F. The process begins at block **1401** in response to receipt by a snooper **304** (e.g., an IMC snooper **126**, L2 cache snooper **116** or a snooper within an I/O controller **128**) of a request. In response to receipt of the request, the snooper **304** determines by reference to the transaction type specified by the request whether or not the request is a write-type request, such as a castout request, write request, or partial write request. In response to the snooper **304** determining at block **1403** that the request is not a write-type request (e.g., a read or RWITM request), the process proceeds to block **1405**, which illustrates the snooper **304** generating the partial response for the request, if required, by conventional processing. If, however, the snooper **304** determines that the request is write-type request, the process proceeds to block **1407**.

Block **1407** depicts the snooper **304** determining whether or not it is the LPC for the request address specified by the write-type request. For example, snooper **304** may make the illustrated determination by reference to one or more base address registers (BARs) and/or address hash functions specifying address range(s) for which the snooper **304** is responsible (i.e., the LPC). If snooper **304** determines that it is not the LPC for the request address, the process passes to block **1409**. Block **1409** illustrates snooper **304** generating a write request partial response **800** (FIG. **8**) in which the valid field **722** and the destination tag field **724** are formed of all '0's, thereby signifying that the snooper **304** is not the LPC for the request address. If, however, snooper **304** determines at block **1407** that it is the LPC for the request address, the process passes to block **1411**, which depicts snooper **304** generating a write request partial response **720** in which valid field **722** is set to '1' and destination tag field **724** specifies a destination tag or route that uniquely identifies the location of snooper **304** within data processing system **200**. Following either of blocks **1409** or **1411**, the process shown in FIG. **14**G ends at block **1413**.

VII. Partial Response Phase Structure and Operation

Referring now to FIG. **15**, there is depicted a block diagram illustrating an exemplary embodiment of the partial response logic **121***b* within interconnect logic **120** of FIG. **1**. As shown, partial response logic **121***b* includes route logic **1500** that routes a remote partial response generated by the snoopers **304** at a remote leaf (or node leaf) **100** back to the remote hub (or node master) **100** from which the request was received via the appropriate one of outbound first tier X, Y and Z links. In addition, partial response logic **121***b* includes combining logic **1502** and route logic **1504**. Combining logic **1502** accumulates partial responses received from remote (or node) leaves **100** with other partial response(s) for the same request that are buffered within NM/RH partial response FIFO queue **940**. For a node-only broadcast operation, the combining logic **1502** of the node master **100** provides the accumulated partial response directly to response logic **122**. For a system-wide broadcast operation, combining logic **1502** supplies the accumulated partial response to route logic **1504**, which routes the accumulated partial response to the local hub **100** via one of outbound A and B links.

Partial response logic **121***b* further includes hold buffers **1506***a*-**1506***b*, which receive and buffer partial responses from remote hubs **100**, a multiplexer **1507**, which applies a fair arbitration policy to select from among the partial responses buffered within hold buffers **1506***a*-**1506***b*, and broadcast logic **1508**, which broadcasts the partial responses selected by multiplexer **1507** to each other processing unit **100** in its processing node **202**. As further indicated by the path coupling the output of multiplexer **1507** to programmable delay **1509**, multiplexer **1507** performs a local broadcast of the partial response that is delayed by programmable delay **1509** by approximately one first tier link latency so that the locally broadcast partial response is received by combining logic **1510** at approximately the same time as the partial responses received from other processing units **100** on the inbound X, Y and Z links. Combining logic **1510** accumulates the partial responses received on the inbound X, Y and Z links and the locally broadcast partial response received from an inbound second tier link with the locally generated partial response (which is buffered within LH partial response FIFO queue **930**) and passes the accumulated partial response to response logic **122** for generation of the combined response for the request.

With reference now to FIG. **16**A-**16**C, there are illustrated flowcharts respectively depicting exemplary processing during the partial response phase of an operation at a native or foreign remote leaf (or for a node-only operation, a node leaf), native or foreign remote hub (or for node-only operations, a node master), and native or foreign local hub. In these figures, transmission of partial responses may be subject to various delays that are not explicitly illustrated. However, because there is no timing constraint on partial response latency as discussed above, such delays, if present, will not induce errors in operation and are accordingly not described further herein.

Referring now specifically to FIG. **16**A, partial response phase processing at a native or foreign remote leaf (or node leaf) **100** begins at block **1600** when the snoopers **304** of the native or foreign remote leaf (or for node-only operations, a node leaf) **100** generate partial responses for the request. As shown at block **1602**, route logic **1500** then routes, using the remote partial response field **712** of the link information allocation, the partial response to the remote hub **100** for the request via the outbound X, Y or Z link corresponding to the inbound first tier link on which the request was received. As indicated above, the inbound first tier link on which the request was received is indicated by which one of tag FIFO queues **924***c***0**-**924***c***2**, **924***d***0**-**924***d***2** and **924***e***0**-**924***e***2** holds master tag for the request. Thereafter, partial response processing continues at the attached native or foreign remote hub (or for node-only operations, the node master) **100**, as indicated by page connector **1604** and as described below with reference to FIG. **16**B.

With reference now to FIG. **16**B, there is illustrated a high level logical flowchart of an exemplary embodiment of a method of partial response processing at a native or foreign remote hub (or for node-only operations, the node master) **100** in accordance with the present invention. The illustrated process begins at page connector **1604** in response to receipt of the partial response of one of the remote leaves (or node leaves) **100** coupled to the remote hub (or node master) **100** by one of the first tier X, Y and Z links. In response to receipt of the partial response, combining logic **1502** reads out the entry **1230** within NM/RH partial response FIFO queue **940** allocated to the operation. The entry is identified by the FIFO ordering observed within NM/RH partial response FIFO queue **940**, as indicated by the X, Y or Z pointer **1216**-**1220** associated with the link on which the partial response was received. Combining logic **1502** then accumulates the partial response of the remote (or node) leaf **100** with the contents of the partial response field **1202** of the entry **1230** that was read.

As mentioned above, the accumulation operation is preferably a non-destructive operation, such as a logical OR operation. Next, combining logic **1502** determines at block **1614** by reference to the response flag array **1234** of the entry **1230** whether, with the partial response received at block **1604**, all of the remote leaves **100** have reported their respective partial responses. If not, the process proceeds to block **1616**, which illustrates combining logic **1502** updating the partial response field **1202** of the entry **1230** allocated to the operation with the accumulated partial response, setting the appropriate flag in response flag array **1234** to indicate which remote leaf **100** provided a partial response, and advancing the associated one of pointers **1216**-**1220**. Thereafter, the process ends at block **1618**.

Referring again to block **1614**, in response to a determination by combining logic **1502** that all attached remote (or node) leaves **100** have reported their respective partial responses for the operation, combining logic **1502** deallocates the entry **1230** for the operation from NM/RH partial response FIFO queue **940** by reference to deallocation pointer **1212**, ending tenure **1308** or **1322** (block **1620**). As indicated by blocks **1621** and **1623**, if the route field **1236** of the entry indicates that the operation is a node-only broadcast operation, combining logic **1502** provides the accumulated partial response directly to response logic **122**. Thereafter, the process passes through page connector **1625** to FIG. **18**A, which is described below. Returning to block **1621**, if the route field **1236** of the deallocated entry indicates that the operation is a system-wide broadcast operation rather than a node-only broadcast operation, combining logic **1502** instead routes the accumulated partial response to the particular one of the outbound A and B links indicated by the contents of route field **1236** utilizing the remote partial response field **712** in the link allocation information, as depicted at block **1622**. Thereafter, the process passes through page connector **1624** to FIG. **16**C.

Referring now to FIG. **16**C, there is depicted a high level logical flowchart of an exemplary method of partial response processing at a native or foreign local hub **100** (including the native or foreign local master **100**) in accordance with an embodiment of the present invention. The process begins at block **1624** in response to receipt at the local hub **100** of a partial response from a remote hub **100** via one of the inbound A and B links. Upon receipt, the partial response is placed within the hold buffer **1506***a*, **1506***b* coupled to the inbound second tier link upon which the partial response was received (block **1626**). As indicated at block **1627**, multiplexer **1507** applies a fair arbitration policy to select from among the partial responses buffered within hold buffers **1506***a*-**1506***b*. Thus, if the partial response is not selected by the fair arbitration policy, broadcast of the partial response is delayed, as shown at block **1628**. Once the partial response is selected by fair arbitration policy, possibly after a delay, multiplexer **1507** outputs the partial response to broadcast logic **1508** and programmable delay **1509**. The output bus of multiplexer **1507** will not become overrun by partial responses because the arrival rate of partial responses is limited by the rate of request launch. Following block **1627**, the process proceeds to block **1629**.

Block **1629** depicts broadcast logic **1508** broadcasting the partial responses selected by multiplexer **1507** to each other processing unit **100** in its processing node **202** via the first tier X, Y and Z links, and multiplexer **1507** performing a local broadcast of the partial response by outputting the partial response to programmable delay **1509**. Thereafter, the process bifurcates and proceeds to each of block **1631**, which illustrates the continuation of partial response phase processing at the other local hubs **100** in the same processing node **202**, and block **1630**. As shown at block **1630**, the partial response broadcast within the present local hub **100** is delayed by programmable delay **1509** by approximately the transmission latency of a first tier link so that the locally broadcast partial response is received by combining logic **1510** at approximately the same time as the partial response(s) received from other processing units **100** on the inbound X, Y and Z links. As illustrated at block **1640**, combining logic **1510** accumulates the locally broadcast partial response with the partial response(s) received from the inbound first tier link and with the locally generated partial response, which is buffered within LH partial response FIFO queue **930**.

In order to accumulate the partial responses, combining logic **1510** first reads out the entry **1200** within LH partial response FIFO queue **930** allocated to the operation. The entry is identified by the FIFO ordering observed within LH partial response FIFO queue **930**, as indicated by the particular one of pointers **1214**, **1215** upon which the partial response was received. Combining logic **1510** then accumulates the locally broadcast partial response of the remote hub **100** with the contents of the partial response field **1202** of the entry **1200** that was read. Next, as shown at blocks **1642**, combining logic **1510** further determines by reference to the response flag array **1204** of the entry **1200** whether or not, with the currently received partial response(s), partial responses have been received from each processing unit **100** from which a partial response was expected.

If not, and if the present local hub is the foreign local master **100**, combining logic **1510** further determines if all partial responses have been received from each processing unit **100** from which a partial response was expected, except the native local hub **100** to which the foreign local master **100** is coupled by one of its second tier links. If so, the foreign local master **100** routes the collected partial response of the foreign plane to a native local hub **100** via one of its second tier A and B links (block **1645**). Thereafter, the process passes through page connect **1624** to block **1626** and following blocks, representing processing of partial responses at the native local hub **100**. In response to a negative determination at block **1643**, the process passes to block **1644**, which depicts combining logic **1510** of the local hub **100** updating the entry **1200** read from LH partial response FIFO queue **930** with the newly accumulated partial response. Thereafter, the process ends at block **1646**.

Returning to block **1642**, if combining logic **1510** determines that all processing units **100** from which partial responses are expected have reported their partial responses, the process proceeds to block **1650**. Block **1650** depicts combining logic **1510** deallocating the entry **1200** allocated to the operation from LH partial response FIFO queue **930** by reference to deallocation pointer **1212**, ending tenure **1304**. Following block **1650**, the process bifurcates and proceeds to each of blocks **1647** and **1652**. Block **1652** depicts combining logic **1510** passes the accumulated partial response read from LH partial response FIFO queue **930** to response logic **122** for generation of the combined response. Thereafter, the process passes through page connector **1654** to FIG. **18**A, which illustrates combined response processing at the local hub **100**. Referring to block **1647**, if the local hub **100** is the native local hub **100** responsible for communication with the foreign plane, the native local hub **100** also transmits the collected partial response of the native plane (which includes the collected partial response of the foreign plane) to the foreign local master **100** via one of its second tier A and B links (block **1649**). Thereafter, the process passes through page connector

1624 to block **1626** and following blocks, representing continued partial response phase processing at the foreign local master **100**.

Referring now to block **1632**, processing of partial response(s) received by a local hub **100** on one or more first tier links begins when the partial response(s) is/are received by combining logic **1510**. As shown at block **1634**, combining logic **1510** may apply small tuning delays to the partial response(s) received on the inbound first tier links in order to synchronize processing of the partial response(s) with each other and the locally broadcast partial response. Thereafter, the partial response(s) are processed as depicted at block **1640** and following blocks, which have been described.

VIII. Combined Response Phase Structure and Operation

Referring now to FIG. **17**, there is depicted a block diagram of exemplary embodiment of the combined response logic **121***c* within interconnect logic **120** of FIG. **1** in accordance with the present invention. As shown, combined response logic **121***c* includes hold buffers **1702***a*-**1702***b*, which each receives and buffers combined responses from a remote hub **100** coupled to the local hub **100** by a respective one of inbound A and B links. The outputs of hold buffers **1702***a*-**1702***b* form two inputs of a first multiplexer **1704**, which applies a fair arbitration policy to select from among the combined responses, if any, buffered by hold buffers **1702***a*-**1702***b* for launch onto first bus **1705** within a combined response field **710** of an information frame.

First multiplexer **1704** has a third input by which combined responses of node-only broadcast operations are presented by response logic **122** for selection and launch onto first bus **1705** within a combined response field **710** of an information frame in the absence of any combined response in hold buffers **1702***a*-**1702***b*. Because first multiplexer **1704** always gives precedence to combined responses for system-wide broadcast operations received from remote hubs **100** over locally generated combined responses for node-only broadcast operations, response logic **122** may, under certain operating conditions, have to wait a significant period in order for first multiplexer **1704** to select the combined response it presents. Consequently, in the worst case, response logic **122** must be able to queue a number of combined response and partial response pairs equal to the number of entries in NM tag FIFO queue **924***b***2**, which determines the maximum number of node-only broadcast operations that a given processing unit **100** can have in flight at any one time. Even if the combined responses are delayed for a significant period, the observation of the combined response by masters **300** and snoopers **304** will be delayed by the same amount of time. Consequently, delaying launch of the combined response does not risk a violation of the timing constraint set forth above because the time between observation of the combined response by the winning master **300** and observation of the combined response by the owning snooper **304** is not thereby decreased.

First bus **1705** is coupled to each of the outbound X, Y and Z links and a node master/remote hub (NM/RH) buffer **1706**. For node-only broadcast operations, NM/RH buffer **1706** buffers a combined response and accumulated partial response (i.e., destination tag) provided by the response logic **122** at this node master **100**.

The inbound first tier X, Y and Z links are each coupled to a respective one of remote leaf (RL) buffers **1714***a*-**1714***c*. The outputs of NM/RH buffer **1706** and RL buffers **1714***a*-**1714***c* form 4 inputs of a second multiplexer **1720**. Second multiplexer **1720** has an additional fifth input coupled to the output of a local hub (LH) hold buffer **1710** that, for a system-wide broadcast operation, buffers a combined response and accumulated partial response (i.e., destination tag) provided by the response logic **122** at this local hub **100**. The output of second multiplexer **1720** drives combined responses onto a second bus **1722** to which tag FIFO queues **924** and the outbound second tier links are coupled. As illustrated, tag FIFO queues **924** are further coupled to receive, via an additional channel, an accumulated partial response (i.e., destination tag) buffered in LH hold buffer **1710** or NM/RH buffer **1706**. Masters **300** and snoopers **304** are further coupled to tag FIFO queues **924**. The connections to tag FIFO queues **924** permits snoopers **304** to observe the combined response and permits the relevant master **300** to receive the combined response and destination tag, if any.

Without the window extension **312***b* described above, observation of the combined response by the masters **300** and snoopers **304** at substantially the same time could, in some operating scenarios, cause the timing constraint term regarding the combined response latency from the winning master **300** to snooper **304***n* (i.e., C_lat(WM_S)) to approach zero, violating the timing constraint. However, because window extension **312***b* has a duration of approximately the first tier link transmission latency, the timing constraint set forth above can be satisfied despite the substantially concurrent observation of the combined response by masters **300** and snoopers **304**.

With reference now to FIG. **18**A-**18**C, there are depicted high level logical flowcharts respectively depicting exemplary combined response phase processing at a native or foreign local hub (or for node-only operations, the node master), native or foreign remote hub (or for node-only operations, the node master), and native or foreign remote leaf (or for node-only operations, node leaf) in accordance with an exemplary embodiment of the present invention. Referring now specifically to FIG. **18**A, combined response phase processing at the native or foreign local hub (or node master) **100** begins at block **1800** and then proceeds to block **1802**, which depicts response logic **122** generating the combined response for an operation based upon the type of request and the accumulated partial response. As indicated at blocks **1803**-**1805**, if the scope indicator **730** within the combined response **710** indicates that the operation is a node-only broadcast operation, combined response phase processing at the node master **100** continues at block **1863** of FIG. **18**B. However, if the scope indicator **730** indicates that the operation is a system-wide broadcast operation, response logic **122** of the remote hub **100** places the combined response and the accumulated partial response into LH hold buffer **1710**, as shown at block **1804**. By virtue of the accumulation of partial responses utilizing an OR operation, for write-type requests, the accumulated partial response will contain a valid field **722** set to '1' to signify the presence of a valid destination tag within the accompanying destination tag field **724**. For other types of requests, bit **0** of the accumulated partial response will be set to '0' to indicate that no such destination tag is present.

As depicted at block **1844**, second multiplexer **1720** is time-slice aligned with the selected second tier link information allocation and selects a combined response and accumulated partial response from LH hold buffer **1710** for launch only if an address tenure is then available for the combined response in the outbound second tier link information allocation. Thus, for example, second multiplexer **1720** outputs a combined response and accumulated partial response from LH hold buffer **1710** only during cycle **1** or **3** of the embodiment of FIG. **7**B. If a negative determination is made at block **1844**, the launch of the combined response within LH hold buffer **1710** is delayed, as indicated at block **1846**, until a subsequent cycle during which an address tenure is available. If, on the other hand, a positive determination is made at block **1844**, second multiplexer **1720** preferentially selects the combined response within LH hold buffer **1710** over its other inputs for launch onto second bus **1722** and subsequent transmission on the outbound second tier links.

It should also be noted that the other ports of second multiplexer **1720** (e.g., RH, RLX, RLY, and RLZ) could also present requests concurrently with LH hold buffer **1710**, meaning that the maximum bandwidth of second bus **1722** must equal 10/8 (assuming the embodiment of FIG. 7B) of the bandwidth of the outbound second tier links in order to keep up with maximum arrival rate. It should further be observed that only combined responses buffered within LH hold buffer **1710** are transmitted on the outbound second tier links and are required to be aligned with address tenures within the link information allocation. Because all other combined responses competing for issuance by second multiplexer **1720** target only the local masters **300**, snoopers **304** and their respective FIFO queues rather than the outbound second tier links, such combined responses may be issued in the remaining cycles of the information frames. Consequently, regardless of the particular arbitration scheme employed by second multiplexer **1720**, all combined responses concurrently presented to second multiplexer **1720** are guaranteed to be transmitted within the latency of a single information frame.

Following the issuance of the combined response on second bus **1722**, the process bifurcates and proceeds to each of blocks **1848** and **1852**. Block **1848** depicts routing the combined response launched onto second bus **1722** to the outbound second tier links for transmission to the remote hubs **100**. Thereafter the process proceeds through page connector **1850** to FIG. **18**C, which depicts an exemplary method of combined response processing at the remote hubs **100**.

Referring now to block **1852**, the combined response issued on second bus **1722** is also utilized to query LH tag FIFO queue **924***a* to obtain the master tag from the oldest entry therein. Thereafter, LH tag FIFO queue **924***a* deallocates the entry allocated to the operation, ending tenure **1302** (block **1854**). Following block **1854**, the process bifurcates and proceeds to each of blocks **1810** and **1856**. At block **1810**, LH tag FIFO queue **924***a* determines whether the master tag indicates that the master **300** that originated the request associated with the combined response resides in this local hub **100**. If not, processing in this path ends at block **1816**. If, however, the master tag indicates that the originating master **300** resides in the present local hub **100**, LH tag FIFO queue **924***a* routes the master tag, the combined response and the accumulated partial response to the originating master **300** identified by the master tag (block **1812**). In response to receipt of the combined response and master tag, the originating master **300** processes the combined response, and if the corresponding request was a write-type request, the accumulated partial response (block **1814**).

For example, if the combined response indicates "success" and the corresponding request was a read-type request (e.g., a read, DClaim or RWITM request), the originating master **300** may update or prepare to receive a requested memory block. In this case, the accumulated partial response is discarded. If the combined response indicates "success" and the corresponding request was a write-type request (e.g., a castout, write or partial write request), the originating master **300** extracts the destination tag field **724** from the accumulated partial response and utilizes the contents thereof as the data tag **714** used to route the subsequent data phase of the operation to its destination. If a "success" combined response indicates or implies a grant of HPC status for the originating master **300**, then the originating master **300** will additionally begin to protect its ownership of the memory block, as depicted at reference numerals **313** and **1314**. If, however, the combined response received at block **1814** indicates another outcome, such as "retry", the originating master **300** may be required to reissue the request, perhaps with a different scope (e.g., global rather than local). Thereafter, the process ends at block **1816**.

Referring now to block **1856**, LH tag FIFO queue **924***a* also routes the combined response and the associated master tag to the snoopers **304** within the local hub **100**. In response to receipt of the combined response, snoopers **304** process the combined response and perform any operation required in response thereto (block **1857**). For example, a snooper **304** may source a requested memory block to the originating master **300** of the request, invalidate a cached copy of the requested memory block, etc. If the combined response includes an indication that the snooper **304** is to transfer ownership of the memory block to the requesting master **300**, snooper **304** appends to the end of its protection window **312***a* a programmable-length window extension **312***b*, which, for the illustrated topology, preferably has a duration of approximately the latency of one chip hop over a first tier link (block **1858**). Of course, for other data processing system topologies and different implementations of interconnect logic **120**, programmable window extension **312***b* may be advantageously set to other lengths to compensate for differences in link latencies (e.g., different length cables coupling different processing nodes **202**), topological or physical constraints, circuit design constraints, or large variability in the bounded latencies of the various operation phases. Thereafter, combined response phase processing at the local hub **100** ends at block **1859**.

Referring now to FIG. **18**B, there is depicted a high level logical flowchart of an exemplary method of combined response phase processing at a native or foreign remote hub (or for node-only operations, the node master) **100** in accordance with the present invention. As depicted, for combined response phase processing at a remote hub **100**, the process begins at page connector **1860** upon receipt of a combined response at a remote hub **100** on one of its inbound A or B links. The combined response is then buffered within the associated one of hold buffers **1702***a*-**1702***b*, as shown at block **1862**. The buffered combined response is then transmitted by first multiplexer **1704** on first bus **1705** as soon as the conditions depicted at blocks **1864** and **1865** are both met. In particular, an address tenure must be available in the first tier link information allocation (block **1864**) and the fair allocation policy implemented by first multiplexer **1704** must select the hold buffer **1702***a*, **1702***b* in which the combined response is buffered (block **1865**).

As shown at block **1864**, if either of these conditions is not met, launch of the combined response by first multiplexer **1704** onto first bus **1705** is delayed until the next address tenure. If, however, both conditions illustrated at blocks **1864** and **1865** are met, the process proceeds from block **1865** to block **1868**, which illustrates first multiplexer **1704** broadcasting the combined response on first bus **1705** to the outbound X, Y and Z links and NM/RH hold buffer **1706** within a combined response field **710**. As indicated by the connection of the path containing blocks **1863** and **1867** to block **1868**, for node-only broadcast operations, first multiplexer **1704** issues the combined response presented by response logic **122** onto first bus **1705** for routing to the outbound X, Y and Z links and NM/RH hold buffer **1706** only if no competing combined responses are presented by hold buffers **1702***a*-**1702***b*. If any competing combined response is received for a system-wide broadcast operation from a remote hub **100** via one of the inbound second tier links, the locally generated combined response for the node-only broadcast operation is delayed, as shown at block **1867**. When first multiplexer **1704** finally selects the locally generated combined response for the node-only broadcast operation, response logic **122** places the associated accumulated partial response directly into NM/RH hold buffer **1706**.

Following block **1868**, the process bifurcates. A first path passes through page connector **1870** to FIG. **18**C, which illustrates an exemplary method of combined response phase processing at the remote leaves (or node leaves) **100**. The second path from block **1868** proceeds to block **1874**, which illustrates the second multiplexer **1720** determining which of the combined responses presented at its inputs to output onto second bus **1722**. As indicated, second multiplexer **1720** prioritizes local hub combined responses over remote hub combined responses, which are in turn prioritized over combined responses buffered in remote leaf buffers **1714***a*-**1714***c*. Thus, if a local hub combined response is presented for selection by LH hold buffer **1710**, the combined response buffered within remote hub buffer **1706** is delayed, as shown at block **1876**. If, however, no combined response is presented by LH hold buffer **1710**, second multiplexer **1720** issues the combined response from NM/RH buffer **1706** onto second bus **1722**.

In response to detecting the combined response on second bus **1722**, the particular one of tag FIFO queues **924***b***0** and **924***b***1** associated with the second tier link upon which the combined response was received (or for node-only broadcast operations, NM tag FIFO queue **924***b***2**) reads out the master tag specified by the relevant request from the master tag field **1100** of its oldest entry, as depicted at block **1878**, and then deallocates the entry, ending tenure **1306** or **1320** (block **1880**). The process then bifurcates and proceeds to each of blocks **1882** and **1881**. Block **1882** depicts the relevant one of tag FIFO queues **924***b* routing the combined response and the master tag to the snoopers **304** in the remote hub (or node master) **100**. In response to receipt of the combined response, the snoopers **304** process the combined response (block **1884**) and perform any required operations, as discussed above. If the operation is a system-wide broadcast operation and if the combined response includes an indication that the snooper **304** is to transfer coherency ownership of the memory block to the requesting master **300**, the snooper **304** appends a window extension **312***b*to its protection window **312***a*, as shown at block **1885**. Thereafter, combined response phase processing at the remote hub **100** ends at block **1886**.

Referring now to block **1881**, if the scope indicator **730** within the combined response field **710** indicates that the operation is not a node-only broadcast operation but is instead a system-wide broadcast operation, no further processing is performed at the remote hub **100**, and the process ends at blocks **1886**. If, however, the scope indicator **730** indicates that the operation is a node-only broadcast operation, the process passes to block **1883**, which illustrates NM tag FIFO queue **924***b***2** routing the master tag, the combined response and the accumulated partial response to the originating master **300** identified by the master tag. In response to receipt of the combined response and master tag, the originating master **300** processes the combined response, and if the corresponding request was a write-type request, the accumulated partial response (block **1887**).

For example, if the combined response indicates "success" and the corresponding request was a read-type request (e.g., a read, DClaim or RWITM request), the originating master **300** may update or prepare to receive a requested memory block. In this case, the accumulated partial response is discarded. If the combined response indicates "success" and the corresponding request was a write-type request (e.g., a castout, write or partial write request), the originating master **300** extracts the destination tag field **724** from the accumulated partial response and utilizes the contents thereof as the data tag **714** used to route the subsequent data phase of the operation to its destination, as described below with reference to FIGS. **20**A-**20**C. If a "success" combined response indicates or implies a grant of HPC status for the originating master **300**, then the originating master **300** will additionally begin to protect its ownership of the memory block, as depicted at reference numerals **313** and **1314**. If, however, the combined response received at block **1814** indicates another outcome, such as "retry", the originating master **300** may be required to reissue the request. Thereafter, the process ends at block **1886**.

With reference now to FIG. **18**C, there is illustrated a high level logical flowchart of an exemplary method of combined response phase processing at a native or foreign remote leaf (or for node-only operations, a node leaf) **100** in accordance with the present invention. As shown, the process begins at page connector **1888** upon receipt of a combined response at the remote (or node) leaf **100** on one of its inbound X, Y and Z links. As indicated at block **1890**, the combined response is latched into one of NL/RL hold buffers **1714***a*-**1714***c*. Next, as depicted at block **1891**, the combined response is evaluated by second multiplexer **1720** together with the other combined responses presented to its inputs. As discussed above, second multiplexer **1720** prioritizes local hub combined responses over remote hub combined responses, which are in turn prioritized over combined responses buffered in NL/RL hold buffers **1714***a*-**1714***c*. Thus, if a local hub or remote hub combined response is presented for selection, the combined response buffered within the NL/RL hold buffer **1714** is delayed, as shown at block **1892**. If, however, no higher priority combined response is presented to second multiplexer **1720**, second multiplexer **920** issues the combined response from the NL/RL hold buffer **1714** onto second bus **1722**.

In response to detecting the combined response on second bus **1722**, the particular one of tag FIFO queues **924***c***0**-**924***c***2**, **924***d***0**-**924***d***2**, and **924***e***0**-**924***e***2** associated with the scope of the operation and the route by which the combined response was received reads out from the master tag field **1100** of its oldest entry the master tag specified by the associated request, as depicted at block **1893**. That is, the scope indicator **730** within the combined response field **710** is utilized to determine whether the request is of node-only or system-wide scope. For node-only broadcast requests, the particular one of NL tag FIFO queues **924***c***2**, **924***d***2** and **924***e***2** associated with the inbound first tier link upon which the combined response was received buffers the master tag. For system-wide broadcast requests, the master tag is retrieved from the particular one of RL tag FIFO queues **924***c***0**-**924***c***1**, **924***d***0**-**924***d***1** and **924***e***0**-**924***e***1** corresponding to the combination of inbound first and second tier links upon which the combined response was received.

Once the relevant tag FIFO queue **924** identifies the appropriate entry for the operation, the tag FIFO queue **924** deallocates the entry, ending tenure **1310** or **1324** (block **1894**). The combined response and the master tag are further routed to the snoopers **304** in the remote (or node) leaf **100**, as shown at block **1895**. In response to receipt of the combined response, the snoopers **304** process the combined response (block **1896**) and perform any required operations, as discussed above. If the operation is not a node-only operation and if the combined response includes an indication that the snooper **304** is to transfer coherency ownership of the memory block to the requesting master **300**, snooper **304** appends to the end of its protection window **312***a* (also protection window **1312** of FIG. **13**) a window extension **312***b*, as described above and as shown at block **1897**. Thereafter, combined response phase processing at the remote leaf **100** ends at block **1898**.

IX. Data Phase Structure and Operation

Data logic **121***d* and its handling of data delivery can be implemented in a variety of ways. In one preferred embodiment, data logic **121***d* and its operation are implemented as described in detail in the co-pending U.S. Patent Applications incorporated by reference above.

X. Conclusion

As has been described, the present invention provides an improved processing unit, data processing system and interconnect fabric for a data processing system. The inventive data processing system topology disclosed herein increases in interconnect bandwidth with system scale. In addition, a data processing system employing the topology disclosed herein may also be hot upgraded (i.e., pairs of native and foreign plane processing nodes may be added during operation), downgraded (i.e., pairs of native and foreign plane processing nodes maybe removed), or repaired without disruption of communication between processing units in the resulting data processing system through the connection, disconnection or repair of individual processing nodes.

The present invention also advantageously supports the concurrent flow of operations of varying scope (e.g., node-only broadcast mode and a system-wide broadcast scope). As will be appreciated, support for operations of less than system-wide scope advantageously conserves bandwidth on the interconnect fabric and enhances overall system performance. Moreover, by throttling the launch of requests in accordance with the servicing rate of snooping devices in the system, snooper retries of operations are advantageously reduced.

While the invention has been particularly shown as described with reference to a preferred embodiment, it will be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention. For example, although the present invention discloses preferred embodiments in which FIFO queues are utilized to order operation-related tags and partial responses, those skilled in the art will appreciated that other ordered data structures may be employed to maintain an order between the various tags and partial responses of operations in the manner described. In addition, although preferred embodiments of the present invention employ uni-directional communication links, those skilled in the art will understand by reference to the foregoing that bi-directional communication links could alternatively be employed.

What is claimed is:

1. A data processing system, comprising:

a first plane including a first plurality of processing nodes each including multiple processing units and a second plane including a second plurality of processing nodes each including multiple processing units;

a plurality of point-to-point type first tier links, wherein each of said first plurality and second plurality of processing nodes includes one or more of first tier links, and wherein a first tier link within a processing node connects solely a pair of processing units in a same processing node for communication; and

a plurality of point-to-point type second tier links, wherein:

at least a first of said plurality of second tier links connects solely two processing units disposed in different ones of said first plurality of processing nodes;

at least a second of said plurality of second tier links connects solely two processing units disposed in different ones of said second plurality of processing nodes; and

at least a third of said plurality of second tier links solely connects a processing unit in said first plane to a processing unit in said second plane;

wherein:

said processing units include interconnect logic that processes a plurality of concurrently pending broadcast operations of differing broadcast scope, wherein at least a first of said plurality of concurrently pending broadcast operations has a first scope including processing nodes in said first and second planes and a second of said plurality of concurrently pending broadcast operations has a second scope restricted to at least one processing node in a single one of said first and second planes;

said first scope comprises a system-wide scope including all processing units in said data processing system;

said interconnect logic places a scope indicator indicating a broadcast scope in at least a request of each operation among said plurality of concurrently pending broadcast operations;

for an operation of system-wide scope, a native local master processing unit in said first plane distributes said operation to each processing unit in said first plane via particular ones of said first and second tier links, and distributes said operation, via a second tier link, to a foreign local master processing unit in said second plane, wherein said foreign local master processing unit distributes said operation to each other processing unit in said second plane via others of said first and second tier links;

said foreign local master processing unit transmits a collected partial response representing all partial responses of processing units in said second plane to a native local hub processing unit in said first plane;

a native local hub processing unit in said first plane transmits a collected partial response representing all partial responses of processing units in said first plane to said foreign local master processing unit in said second plane; and

said foreign local master processing unit determines a combined response representing a system-wide response to said operation based at least in part upon said collected partial response of said first plane.

2. A data processing system, comprising:

a first plane including a first plurality of processing nodes each including multiple processing units and a second plane including a second plurality of processing nodes each including multiple processing units;

a plurality of point-to-point type first tier links, wherein each of said first plurality and second plurality of processing nodes includes one or more of first tier links, and wherein a first tier link within a processing node connects solely a pair of processing units in a same processing node for communication; and

a plurality of point-to-point type second tier links, wherein:

at least a first of said plurality of second tier links connects solely two processing units disposed in different ones of said first plurality of processing nodes;

at least a second of said plurality of second tier links connects solely two processing units disposed in different ones of said second plurality of processing nodes; and

at least a third of said plurality of second tier links solely connects a processing unit in said first plane to a processing unit in said second plane;

wherein:

at least some of the processing units in the data processing system have associated cache memory;

the data processing system is cache coherent; and

wherein for an operation of system-wide scope, a native local master processing unit in said first plane distributes said operation to each processing unit in said first plane via particular ones of said first and second tier links, and distributes said operation, via a second tier link, to a foreign local master processing unit in said second plane, wherein said foreign local master processing unit distributes said operation to each other processing unit in said second plane via others of said first and second tier links.

3. The data processing system of claim **2**, wherein:

said processing units include interconnect logic that processes a plurality of concurrently pending broadcast operations of differing broadcast scope, wherein at least a first of said plurality of concurrently pending broadcast operations has a first scope including processing nodes in said first and second planes and a second of said plurality of concurrently pending broadcast operations has a second scope restricted to at least one processing node in a single one of said first and second planes.

4. The data processing system of claim **3**, wherein said first scope comprises a system-wide scope including all processing units in said data processing system.

5. The data processing system of claim **3**, wherein said interconnect logic places a scope indicator indicating a broadcast scope in at least a request of each operation among said plurality of concurrently pending broadcast operations.

6. The data processing system of claim **2**, wherein said at least a third of said plurality of second tier links includes multiple second tier links each connecting a respective one of said plurality of first processing nodes in said first plane to a single one of said plurality of second processing nodes in said second plane.

7. The data processing system of claim **2**, wherein said foreign local master processing unit transmits a collected partial response representing all partial responses of processing units in said second plane to a native local hub processing unit in said first plane.

8. The data processing system of claim **7**, wherein a native local hub processing unit in said first plane transmits a collected partial response representing all partial responses of processing units in said first plane to said foreign local master processing unit in said second plane.

9. The data processing system of claim **8**, wherein said foreign local master processing unit determines a combined response representing a system-wide coherence response to said operation based at least in part upon said collected partial response of said first plane.

10. A method of data processing in a data processing system including a first plane containing a first plurality of processing nodes each including multiple processing units and a second plane containing a second plurality of processing nodes each including multiple processing units, said method comprising:

communicating operations between processing units within a same processing node via a plurality of point-

to-point first tier links, wherein each of said first plurality and second plurality of processing nodes includes one or more first tier links among said plurality of first tier links, and wherein each first tier link connects solely a pair of processing units in a same processing node for communication; and

communicating operations between processing units in different processing nodes via a plurality of point-to-point second tier links, wherein:

at least a first of said plurality of second tier links connects solely two processing units in different ones of said first plurality of processing nodes;

at least a second of said plurality of second tier links connects solely processing units in different ones of said second plurality of processing nodes; and

at least a third of said plurality of second tier links connects solely a processing unit in said first plane to a processing unit in said second plane;

wherein:

at least some of the processing units in the data processing system have associated cache memory;

the data processing system is cache coherent; and said steps of communicating operations between processing units within a same processing node and communicating operations between processing units in different processing nodes comprise:

transmitting an operation of system-wide scope, wherein said transmitting includes:

a native local master processing unit in said first plane distributing said operation to each processing unit in said first plane via particular ones of said first and second tier links; and

distributing said operation, via a second tier link, to a foreign local master processing unit in said second plane, wherein said foreign local master processing unit distributes said operation to each other processing unit in said second plane via others of said first and second tier links.

11. The method of claim **10**, wherein:

interconnect logic in said processing units processing a plurality of concurrently pending broadcast operations of differing broadcast scope, wherein at least a first of said plurality of concurrently pending broadcast operations has a first scope including processing nodes in said first and second planes and a second of said plurality of concurrently pending broadcast operations has a second scope restricted to at least one processing node in a single one of said first and second planes.

12. The method of claim **11**, wherein said first scope comprises a system-wide scope including all processing units in said data processing system.

13. The method of claim **11**, and further comprising said interconnect logic placing a scope indicator indicating a broadcast scope in at least a request of each operation among said plurality of concurrently pending broadcast operations.

14. The method of claim **10**, wherein:

said at least a third of said plurality of second tier links includes multiple second tier links each connecting a respective one of said plurality of first processing nodes in said first plane to a single one of said plurality of second processing nodes in said second plane; and

said step of communicating operations between processing units in different processing nodes via a plurality of point-to-point second tier links comprises communicating operations between said first and second planes via said multiple second tier links.

15. The method of claim **10**, and further comprising said foreign local master processing unit transmitting a collected partial response representing all partial responses of processing units in said second plane to a native local hub processing unit in said first plane.

16. The method of claim **15**, and further comprising a native local hub processing unit in said first plane transmitting a collected partial response representing all partial responses of processing units in said first plane to said foreign local master processing unit in said second plane.

17. The method of claim **16**, and further comprising said foreign local master processing unit determining a combined response representing a system-wide coherence response to said operation based at least in part upon said collected partial response of said first plane.

* * * * *